

MP1520B
PDH Analyzer
Operation Manual
Vol.2

Read this manual before using the equipment.
Keep this manual with the equipment.



MP1520B
PDH Analyzer
Operation Manual
Vol.2

Remote Control Options
(GPIB: OPT 04 RS-232C: OPT 05)

Fourth Edition

Read this manual before using the equipment.
Keep this manual with the equipment.

Measuring Instruments Division
Measurement Group
ANRITSU CORPORATION

Safety Symbols

To prevent the risk of personal injury or loss related to equipment malfunction, Anritsu Corporation uses the following safety symbols to indicate safety-related information. Insure that you clearly understand the meanings of the symbols BEFORE using the equipment.

Symbols used in manual

DANGER	This indicates a very dangerous procedure that could result in serious injury or death if not performed properly.
WARNING	This indicates a hazardous procedure that could result in serious injury or death if not performed properly.
CAUTION	This indicates a hazardous procedure or danger that could result in light-to-severe injury, or loss related to equipment malfunction, if proper precautions are not taken.

Safety Symbols Used on Equipment and in Manual

(Some or all of the following five symbols may not be used on all Anritsu equipment. In addition, there may be other labels attached to products which are not shown in the diagrams in this manual.) The following safety symbols are used inside or on the equipment near operation locations to provide information about safety items and operation precautions. Insure that you clearly understand the meanings of the symbols and take the necessary precautions BEFORE using the equipment.



This indicates a prohibited operation. The prohibited operation is indicated symbolically in or near the barred circle.



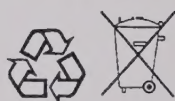
This indicates an obligatory safety precaution. The obligatory operation is indicated symbolically in or near the circle.



This indicates warning or caution. The contents are indicated symbolically in or near the triangle.



This indicates a note. The contents are described in the box.



These indicate that the marked part should be recycled.

MP1520B PDH Analyzer Operation Manual Vol.2
Remote Control Options (GPIB: OPT 04 RS-232C: OPT 05)

January 1995 (First Edition)
February 1996 (Fourth Edition)

Copyright © 1995-1996 by Anritsu Corporation

All rights reserved. No part of this manual may be reproduced without the prior written permission of Anritsu Corporation.

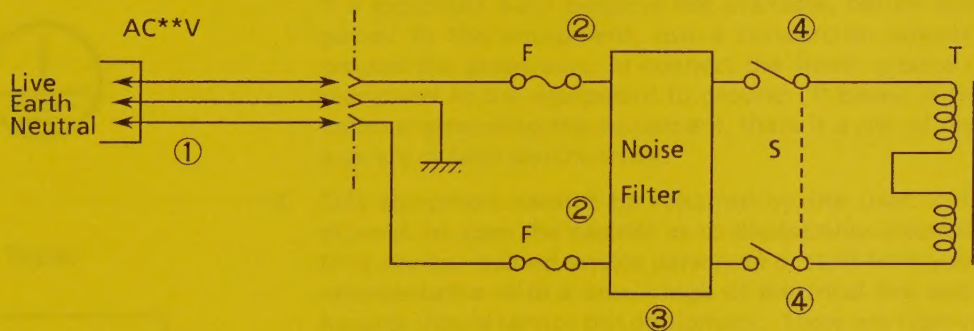
The contents of this manual may be changed without prior notice.

Printed in Japan

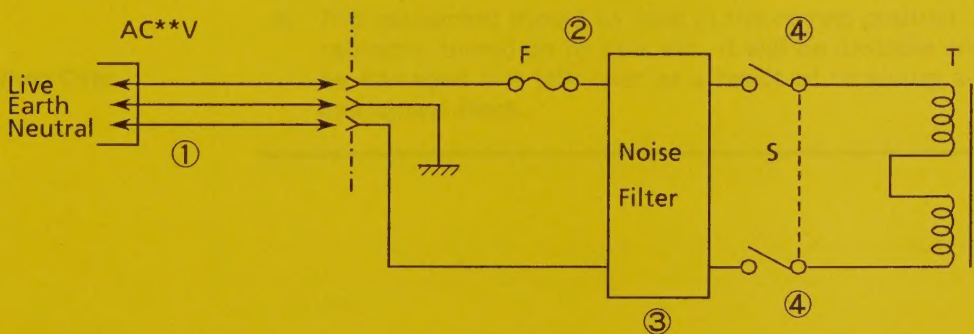
A2. ONE-FUSE MODIFICATION

The Power Supply Circuit for this instrument has been modified as follows. Therefore, when reading this manual, please apply the following information wherever relevant.

1. Description in the manual



2. Modified power supply



Parts being used :

- ① Power cord
- ② One fuse holder with a built-in fuse
- ③ Noise filter
- ④ Power switch

ALTERNATIVE INVESTMENT

The first part of the course is devoted to the study of the various types of alternative investments and their characteristics. This includes a detailed analysis of the different types of alternative investments and their characteristics, as well as a discussion of the various factors that can influence the performance of these investments.

1. Introduction to the course

Topic	Content
1.1. Introduction to the course	Overview of the course and its objectives.
1.2. The role of alternative investments in a portfolio	Discussion of the various factors that can influence the performance of alternative investments.
1.3. The different types of alternative investments	Analysis of the different types of alternative investments and their characteristics.
1.4. The importance of due diligence	Discussion of the various factors that can influence the performance of alternative investments.
1.5. The importance of diversification	Discussion of the various factors that can influence the performance of alternative investments.

2. The different types of alternative investments

Type of investment	Characteristics
2.1. Private equity	Investment in private companies.
2.2. Venture capital	Investment in early-stage companies.
2.3. Real estate	Investment in physical property.
2.4. Hedge funds	Investment in a variety of assets.
2.5. Commodity funds	Investment in commodities.

3. The importance of due diligence

1. The importance of due diligence
2. The importance of diversification
3. The importance of risk management
4. The importance of liquidity
5. The importance of transparency

For Safety

WARNING



1. Always refer to the operation manual when working near locations at which the alert mark shown on the left is attached. If the operation, etc., is performed without heeding the advice in the operation manual, there is a risk of personal injury. In addition, the equipment performance may be reduced.

Moreover, this alert mark is sometimes used with other marks and descriptions indicating other dangers.



2. When supplying power to this equipment, connect the accessory 3-pin power cord to a 3-pin grounded power outlet. If a grounded 3-pin outlet is not available, before supplying power to the equipment, use a conversion adapter and ground the green wire, or connect the frame ground on the rear panel of the equipment to ground. If power is supplied without grounding the equipment, there is a risk of receiving a severe or fatal electric shock.

Repair

WARNING

3. This equipment cannot be repaired by the user. DO NOT attempt to open the cabinet or to disassemble internal parts. Only Anritsu-trained service personnel or staff from your sales representative with a knowledge of electrical fire and shock hazards should service this equipment. There are high-voltage parts in this equipment presenting a risk of severe injury or fatal electric shock to untrained personnel. In addition, there is a risk of damage to precision parts.

Falling Over

4. This equipment should be used in the correct position. If the cabinet is turned on its side, etc., it will be unstable and may be damaged if it falls over as a result of receiving a slight mechanical shock.

Anritsu Corporation Contact

If this equipment develops a fault, contact the head office of Anritsu Corporation at the address in the operation manual, or your nearest sales or service office listed on the following pages.

For Safety

CAUTION

Changing Fuse

CAUTION

1. Before changing the fuses, ALWAYS remove the power cord from the poweroutlet and replace the blown fuses. Always use new fuses of the type and rating specified on the fuse marking on the rear panel of the cabinet.

T □ □ □ A indicates a time-lag fuse.

There is risk of receiving a fatal electric shock if the fuses are replaced with the power cord connected.

2. Keep the power supply and cooling fan free of dust.
 - Clean the power inlet regularly. If dust accumulates around the power pins, there is a risk of fire.
 - Keep the cooling fan clean so that the ventilation holes are not obstructed. If the ventilation is obstructed, the cabinet may overheat and catch fire.

Cleaning

Equipment Certificate

Anritsu Corporation certifies that this equipment was tested before shipment using calibrated measuring instruments with direct traceability to public testing organizations recognized by national research laboratories including the Electrotechnical Laboratory, the National Research Laboratory and the Communication Research laboratory, and was found to meet the published specifications.

Anritsu Warranty

Anritsu Corporation will repair this equipment free-of-charge if a malfunction occurs within 1 year after shipment due to a manufacturing fault, provided that this warranty is rendered void under any or all of the following conditions.

- The fault is outside the scope of the warranty conditions described in the operation manual.
- The fault is due to misoperation, misuse, or unauthorized modification or repair of the equipment by the customer.
- The fault is due to severe usage clearly exceeding normal usage.
- The fault is due to improper or insufficient maintenance by the customer.
- The fault is due to natural disaster including fire, flooding and earthquake, etc.
- The fault is due to use of non-specified peripheral equipment, peripheral parts, consumables, etc.
- The fault is due to use of a non-specified power supply or in a non-specified installation location.

In addition, this warranty is valid only for the original equipment purchaser. It is not transferable if the equipment is resold.

Anritsu Corporation will not accept liability for equipment faults due to unforeseen and unusual circumstances, nor for faults due to mishandling by the customer.

Anritsu Corporation Contact

If this equipment develops a fault, contact the head office of Anritsu Corporation at the address in the operation manual, or your nearest sales or service office listed on the following pages.

vi

MEMORY BACK-UP BATTERY REPLACEMENT

The power for memory back-up is supplied by a Poly-carbomono-fluoride Lithium Battery. This battery should only be replaced by a battery of the same type; since replacement can only be made by Anritsu, contact the nearest Anritsu representative when replacement is required.

STORAGE MEDIUM

This equipment stores data and programs using backed - up memories . Data and programs may be lost due to improper use or failure.

ANRITSU CANNOT COMPENSATE FOR ANY MEMORY LOSS.

Please pay careful attention to the following points.

(Backed-up memory)

- Isolate the memory from static electricity.

Note: *The battery life is about 7 years. Early battery replacement is recommended.*

CONCERNING THE DISPOSAL OF AN MP1520B

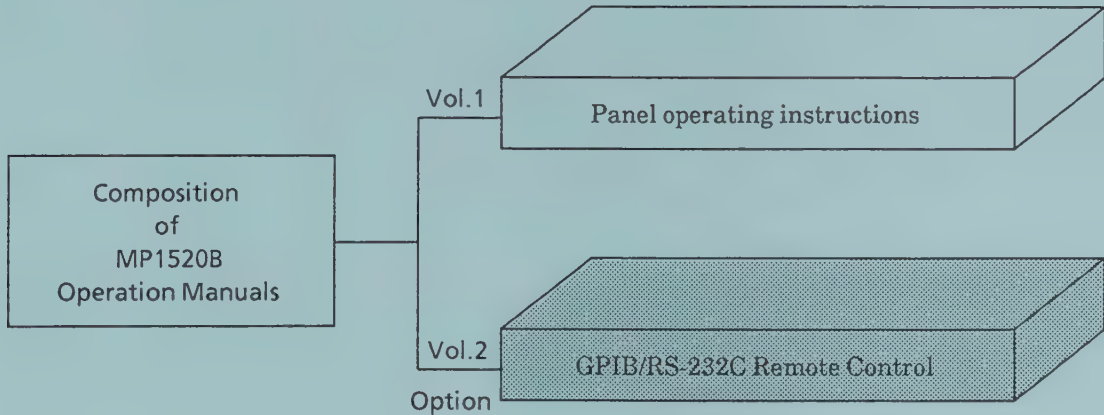
This instrument employs LEDs which contain an arsenical compound. Should this instrument be discarded for any reason, an adequate care should be taken so that it is disposed of according to the waste disposal laws of your own country.

(Blank)

ABOUT THIS MANUAL

Composition of MP1520B Operation Manuals

The MP1520B PDH Analyzer Operation Manuals are composed of the following two documents. Use them properly according to the usage purpose.



Panel operating instructions:

Provides an outline of the MP1520B, and describes preparation before use, manual operation (local operation: panel description, basic operating instructions, detailed operating instructions), measurement examples, performance tests, and storage and transportation.

GPIB/RS-232C Remote Control:

Describes the remote control and program examples used for GPIB and RS-232C interfaces. This manual is supplied with the GPIB or RS-232C interface unit sold separately as Option 04 or Option 05.

(The MP1520B complies with SCPI.)

TABLE OF CONTENTS

For Safety		iii
SECTION 1	GENERAL INFORMATION	1-1
1.1	Interface Function	1-3
1.2	Examples of System Configuration Using Interfaces	1-4
SECTION 2	GPIB INTERFACE	2-1
2.1	GPIB Interface Function	2-3
2.2	Device Message List	2-4
2.3	Bus Command	2-5
2.4	Installing GPIB Interface	2-6
2.5	Connecting GPIB Cables	2-7
2.6	Setting GPIB	2-8
2.7	Device Initialization	2-9
2.7.1	Bus initialization	2-9
2.7.2	Message exchange initialization	2-9
2.7.3	Device initialization	2-9
2.7.4	Device status at power-on	2-10
SECTION 3	RS-232C INTERFACE	3-1
3.1	Device Message List	3-3
3.2	Installing RS-232C Interface	3-5
3.3	Connecting RS-232C Cable	3-6
3.4	Setting RS-232C	3-7
3.5	Device Initialization	3-9
3.5.1	Device initialization	3-9
3.5.2	Device status at power-on	3-9
SECTION 4	LISTENER INPUT FORMAT	4-1
4.1	Listener Input Program Message Format	4-3
4.2	Functional Elements of Program Messages	4-5
4.2.1	<TERMINATED PROGRAM MESSAGE>	4-5
4.2.2	<PROGRAM MESSAGE TERMINATOR>	4-6
4.2.3	<white space>	4-7
4.2.4	<PROGRAM MESSAGE>	4-7
4.2.5	<PROGRAM MESSAGE UNIT SEPARATOR>	4-8

4.2.6	<PROGRAM MESSAGE UNIT>	4-8
4.2.7	<COMMAND MESSAGE UNIT> and <QUERY MESSAGE UNIT>	4-9
4.2.8	<COMMAND PROGRAM HEADER>	4-10
4.2.9	<QUERY PROGRAM HEADER>	4-12
4.2.10	<PROGRAM HEADER SEPARATOR>	4-13
4.2.11	<PROGRAM DATA SEPARATOR>	4-13
4.3	Program Data Format	4-14
4.3.1	<CHARACTER PROGRAM DATA>	4-15
4.3.2	<DECIMAL NUMERIC PROGRAM DATA>	4-16
4.3.3	<BOOLEAN PROGRAM DATA>	4-17
4.3.4	<STRING PROGRAM DATA>	4-18
SECTION 5	TALKER OUTPUT FORMAT	5-1
5.1	Talker Output Response Message Format	5-3
5.2	Functional Elements of Response Message	5-4
5.2.1	<TERMINATED RESPONSE MESSAGE>	5-4
5.2.2	<RESPONSE MESSAGE TERMINATOR>	5-5
5.2.3	<RESPONSE MESSAGE>	5-6
5.2.4	<RESPONSE MESSAGE UNIT SEPARATOR>	5-6
5.2.5	<RESPONSE MESSAGE UNIT>	5-7
5.2.6	<RESPONSE DATA SEPARATOR>	5-7
5.2.7	<RESPONSE DATA	5-8
SECTION 6	IEEE488.2 COMMON COMMANDS	6-1
SECTION 7	STATUS REPORTING	7-1
7.1	MP1520B Status Register Structure	7-3
7.2	IEEE488.2-Defined Status Register	7-5
7.3	SCPI-Defined Status Registers	7-7
7.4	MP1520B-Dependent Status Register	7-8
7.5	Reading, Writing to, and Clearing Status Registers	7-11
7.5.1	Methods of reading and writing	7-11
7.5.2	Methods of clearing and resetting	7-12
SECTION 8	GENERAL INFORMATION ON SCPI	8-1
8.1	Outline	8-3
8.2	Command Structure	8-3
8.3	Writing Commands	8-4

8.4	Compounding Commands	8-5
8.5	Parameter	8-5
SECTION 9	DEVICE MESSAGES	9-1
9.1	Response Format	9-3
9.2	Buffer Size Requirements	9-6
9.3	Device-dependent Commands	9-7
9.3.1	INSTRument subsystem	9-8
9.3.2	SOURce subsystem	9-9
9.3.3	SENSe subsystem	9-26
9.3.4	DISPlay subsystem	9-43
9.3.5	ROUTe subsystem	9-59
9.3.6	CALCulate subsystem	9-60
9.3.7	SYSTem subsystem	9-70
9.3.8	TEST subsystem	9-82
9.3.9	STATus subsystem	9-84
SECTION 10	PROGRAM EXAMPLES	10-1
10.1	Loop Back Measurement	10-3
10.2	Monitor Measurements	10-10
APPENDIX A	SCPI CONFORMANCE INFORMATION	A-1
A.1	SCPI Version	A-1
A.2	SCPI-defined Commands	A-1
A.3	Non-SCPI Commands	A-2
APPENDIX B	SELF-TEST ERROR CODES	B-1
APPENDIX C	SCPI ERROR MESSAGES	C-1
C.1	Command Errors	C-1
C.2	Execution Errors	C-2
C.3	Device-Specific Errors	C-3
C.4	Query Errors	C-4

APPENDIX D	CORRESPONDENCE BETWEEN SCREENS AND COMMANDS	D-1
D.1	The Correspondence between Condtn Screens and Commands	D-1
D.1.1	Condtn:Tx&Rx screen	D-1
D.1.2	Condtn:Tx screen	D-1
D.1.3	Condtn:Rx screen	D-2
D.2	The Correspondence between Results Screens and Commands	D-2
D.3	The Correspondence between Analysis Screens and Commands	D-3
D.3.1	Analysis:Trace Text screen	D-3
D.3.2	Analysis:Trace Graph screen	D-4
D.3.3	Analysis:Distribution screen	D-4
D.4	The Correspondence between Setup Screens and Commands	D-5
D.4.1	Setup:Measure screen	D-5
D.4.2	Setup:Performance screen	D-5
D.4.3	Setup:Print screen	D-7
D.4.4	Setup:Memory screen	D-7
D.5	The Correspondence between System Sreen and Commands	D-8
D.6	The Correspondence between Self-Test Screen and Commands	D-8
D.7	The Correspondence Jitter Screen and Commands	D-9
D.7.1	Jitter:Manual screen	D-9
D.7.2	Jitter:Sweep mask screen	D-10
D.7.3	Jitter:Transfer screen	D-11
D.7.4	Jitter:Jitt/Freq screen	D-12
D.8	The Correspondence Between Front Panel and Other Commands	D-13
APPENDIX E	DETAILED FUNCTIONS OF INITIALIZATION COMMANDS	E-1
E.1	Bus Initialization	E-1
E.2	Message Exchange Initialization	E-2
E.3	Device Initialization	E-3
E.4	Device Status at Power-on	E-3

(Blank)

SECTION 1

GENERAL INFORMATION

The MP1520B PDH Analyzer, when connected with an external controller, automates your measurement system. To connect with the controller, a GPIB interface or an RS-232C interface is used. This section provides information about the interface functions and examples of system configuration.

TABLE OF CONTENTS

1.1	Interface Function	1-3
1.2	Examples of System Configuration Using Interfaces	1-4

(Blank)

SECTION 1

GENERAL INFORMATION

1.1 Interface Function

The MP1520B PDH Analyzer has one slot for remote control on the rear panel. Equipment can be controlled remotely and data can be printed out from an external printer by plugging either a GPIB board or an RS-232C board into the slot. This instrument can detect automatically whether a GPIB or a RS-232C board is plugged into the slot.

GPIB Interface: The GPIB interface employed in this instrument conforms to the IEEE std 488.1-1987 standard. The software conforms to IEEE488.2 and the SCPI (Standard Commands for Programmable Instruments; for details see Section 9).

RS-232C Interface: Commands for the GPIB interface can also be used with this interface.

This instrument provides the following interface functions.

- Controls all functions except the power switch and some keys including the Local key
- Reads out all setting conditions and screen display contents
- Outputs to an external printer

1.2 Examples of System Configuration Using Interfaces

Examples of system configurations using interfaces are shown below.

(1) Control by the host computer

The host computer and this instrument are connected to configure an automatic measurement system.

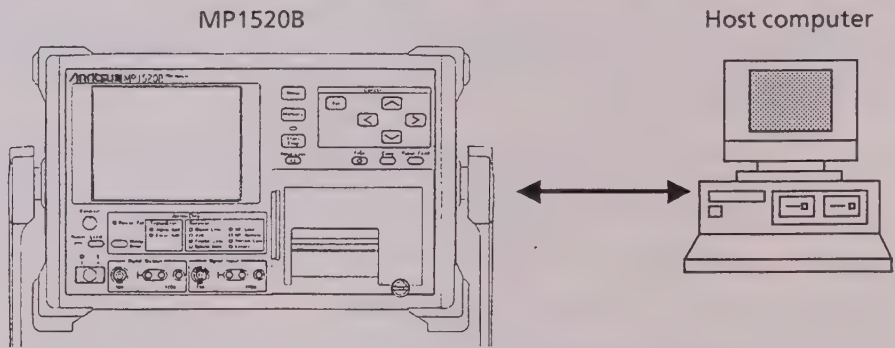


Fig. 1-1 Interface System Configuration (1)

(2) Output to an external printer

When an external printer is connected, the data measured by this instrument can be output to that printer by panel operation.

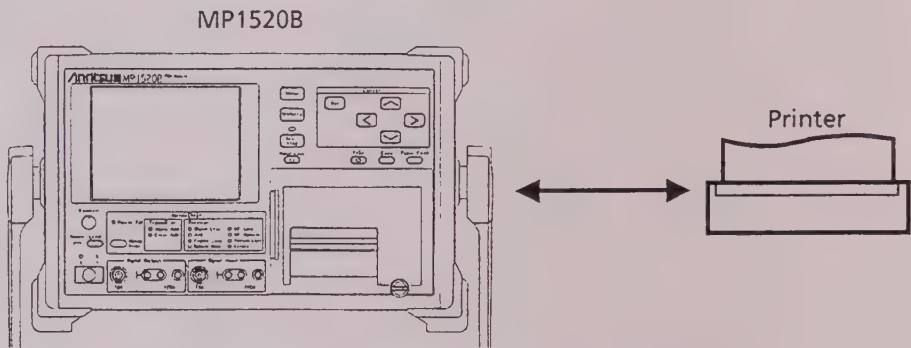


Fig. 1-2 Interface System Configuration (2)

SECTION 2

GPIB INTERFACE

This section describes the GPIB interface functions, methods of setting, and so forth when the GPIB interface is used as an option.

TABLE OF CONTENTS

2.1	GPIB Interface Function	2-3
2.2	Device Message List	2-4
2.3	Bus Command	2-5
2.4	Installing GPIB Interface	2-6
2.5	Connecting GPIB Cables	2-7
2.6	Setting GPIB	2-8
2.7	Device Initialization	2-9
2.7.1	Bus initialization	2-9
2.7.2	Message exchange initialization	2-9
2.7.3	Device initialization	2-9
2.7.4	Device status at power-on	2-10

(Blank)

SECTION 2

GPIB INTERFACE

2.1 GPIB Interface Function

This instrument without controller functions is used for device functions only. Therefore, this instrument has the GPIB interface functions shown in the table below.

Table 2-1 GPIB Interface Functions

Code	Interface function	IEEE488.2 standard
SH1	All source handshake functions	All functions are provided as standard
AH1	All acceptor handshake functions	All functions are provided as standard
T5	Basic talker functions. Serial poll functions. Talk-only functions. A function which can cancel the talker with MLA.	A device must have one of the T5, T6, or TE6 subsets
L3	Basic listener functions. Listener only functions A function which can cancel the listener with MTA.	A device must have one of the L3, LE3, or LE4 subsets
SR1	All service request functions	All functions are provided as standard
RL1	All remote and local control functions	RL0 (no functions) or RL1 (all functions)
PP0	No parallel poll functions	PP0 (no functions) or PP1 (all functions)
DC1	All device clear functions	All functions are provided as standard
DT1	All device trigger functions	DT0 (no functions not provided) or DT1 (all functions provided)
C0	No system controller functions	C0 (no functions) or C4 and C5, or any of C7, C9, C11

2.2 Device Message List

A device message is a data message that is transferred between a controller and a device via the system interface when the bus is in data mode (while the ATN line is at 'H' level.).

Data messages consist of two types: program messages and response messages. A program message is an ASCII data message transferred from the controller to the device. A response message is a data message sent from the device to the controller.

Some types of program and response messages are shown below.

Table 2-2 Device Message List

Program message (For details see Section 4)	Response message (For details see Section 5)
Program command • Device-dependent specific command (For details see Section 9) • IEEE488.2 common command (For details see Section 6) Program query	Status message (for details see Section 7) Response message

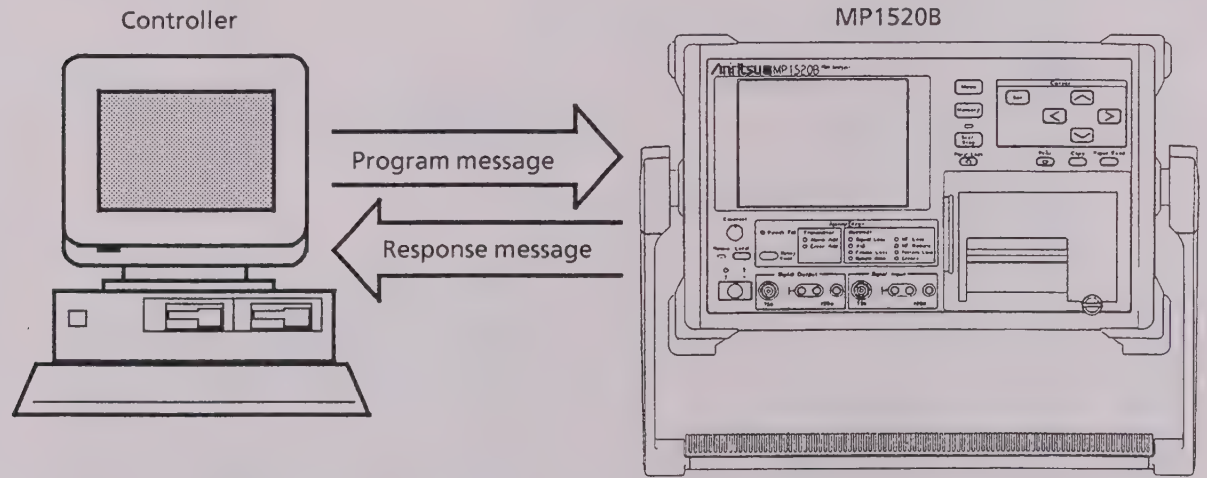


Fig. 2-1 Device Messages

The messages described above are transferred via the input and output buffers of the device. The table below briefly describes the input and output buffers.

Table 2-3 Input and Output Buffers

Input buffer	Output buffer
A FIFO (first In First Out) memory area where DAB (program messages or query messages) are stored temporarily before they are parsed. The input buffer size of this instrument is 256 bytes.	A FIFO queue memory area. All DAB (response messages) output from a device to the controller are stored in this area until read by the controller.

2.3 Bus Command

A bus command is used to manage internal communication on the interface bus when the bus is in command mode (while the ATN line is at 'L' level.).

Bus commands are listed in the following table.

Table 2-4 Contents of Bus Commands

Bus command	Operation
DCL (Device clear)	Executes the initialization for message exchange of all devices connected to the GPIB bus
SDC (Selected Device Clear)	Executes the initialization for message exchange of the device whose address is specified . This command is identical to the DCL command in operation.
GET (Group Execute Trigger)	Operates in the same way when the Start key has been pressed
IFC (Interface Clear)	Initializes the interface bus

2.4 Installing GPIB Interface

Use the GPIB interface (OPT04 Plug-in Unit) by plugging the Plug-in Unit into the slot as shown below

CAUTION

When replacing the plug-in unit, be careful not to injure your fingers.

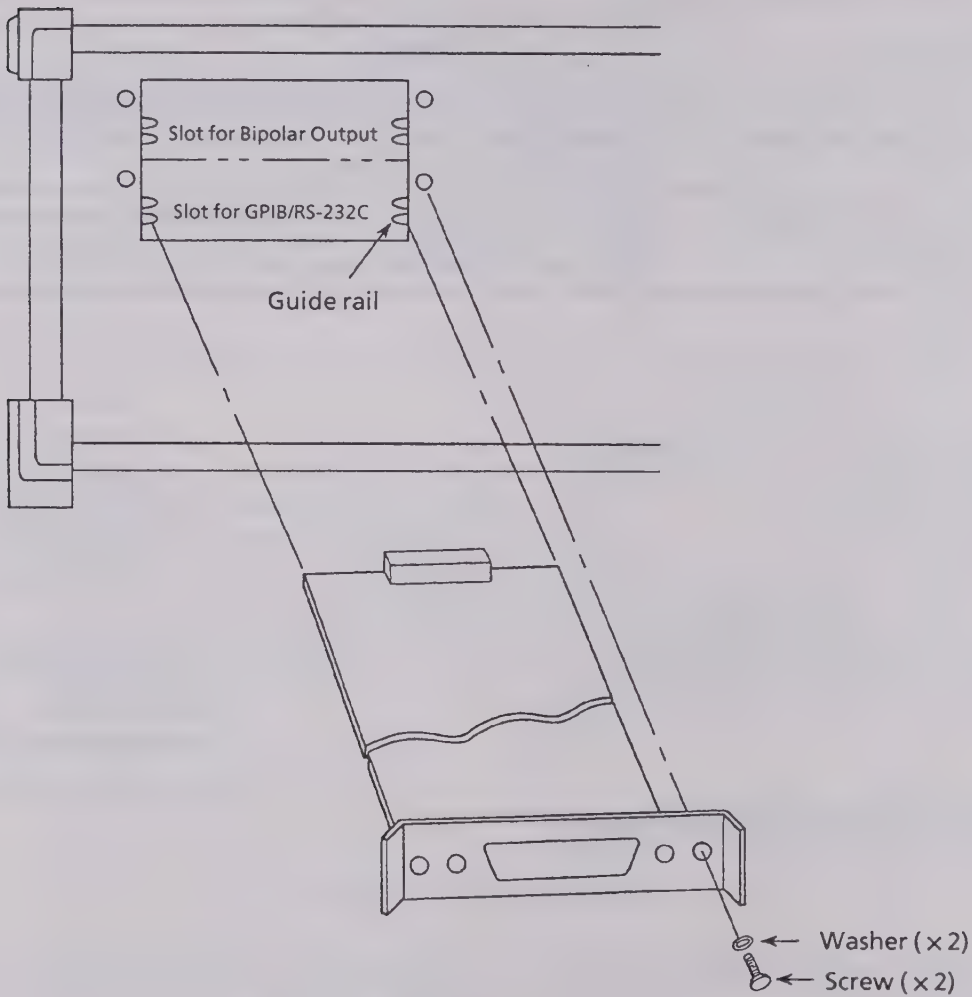


Fig. 2-2 Plugging in the Plug-in Unit

Step	Procedure
1	Check that the power switch has been turned OFF before replacing the plug-in unit.
2	Check which slot position to use in Fig. 2-3 before plugging in the plug-in unit.
3	To plug the plug-in unit into the slot, fit the unit end accurately to the two guide rails inside the slot, and insert the unit to the inner part so that its connector can be mated firmly, then tighten the right-to-left screws. Plugging in the unit or tightening the screws incorrectly may cause faulty operation.
4	Although a slot which does not contain a plug-in unit may not affect the operation of this instrument, it is recommended that such a slot be covered with a blank panel.

2.5 Connecting GPIB Cables

Connect the GPIB cables to the GPIB connector on the MP1520B rear panel as shown in Fig.2-3. The GPIB system configuration is restricted as follows:

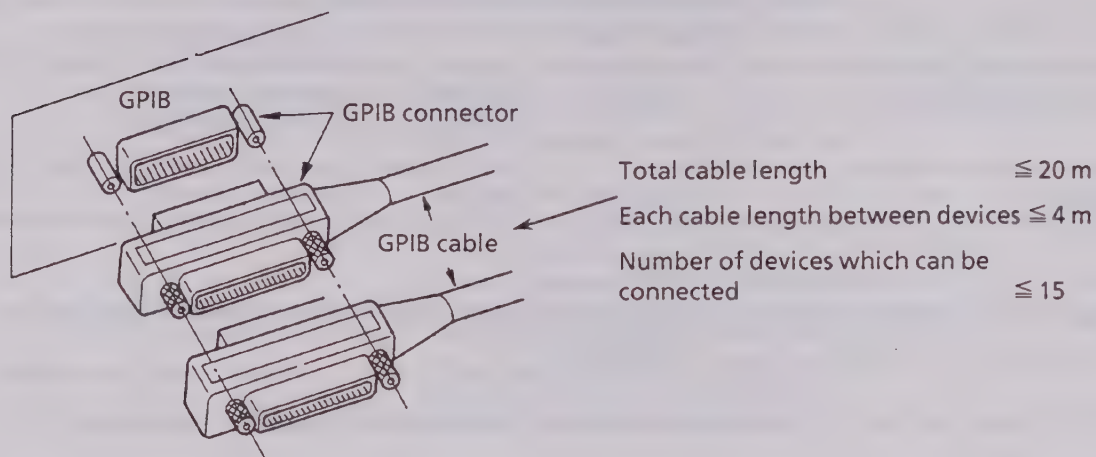


Fig. 2-3 GPIB Cable Connection

2.6 Setting GPIB

To use GPIB as a remote interface, place the instrument in local mode and then make the settings shown below using the front-panel keys.

Table 2-5 Setting Contents in GPIB Interface Use

Setting contents	Setting item	Setting range
How to use interface	Usage	Control: Used for remote control Printer: Used for external printer
Address	Address	0 to 30

Make the above settings using the System screen.

Step	Procedure
1	(1) Press the Menu key, and a window as shown below opens in the upper-right corner of the screen. (2) Use the Cursor key to position the window cursor on the item labeled System then press the Set key (or Menu key), and the current screen is switched to the System screen. (1) Condtn/Result Analysis Setup System Selftest (2) Condtn/Result Analysis Setup System Selftest

Fig. 2-4 Screen Selection Window

- 2
- The above operation at Step 1 causes transition to the System screen shown below.
Move the reverse cursor to the *1 or *2 position to make appropriate settings. For details of settings, refer to the panel operating instruction section of the separate Operation Manual.

System		
Buzzer	:Error/Alarm	
Date & Time	:93-10-16 10:00:00	
GPIB interface		
Usage	:Control	*1
Address	:15	*2

Fig. 2-5 System Screen (While GPIB interface is being operated)

2.7 Device Initialization

The IEEE488.2 standard stipulates three levels for system initialization. The first level is 'bus initialization', the second level is 'message exchange initialization', and the third level is 'device initialization'.

Table 2-6 System Initialization Types

Level	Initialization type	Outline
1	Bus initialization	Initializes all interface functions connected to the bus by the IFC message from the controller.
2	Message exchange initialization	Invalidates the functions to report the operation end to the controller, or initializes the message exchange of all devices or specified devices, when the DCL or SDC GPIB bus commands respectively are executed.
3	Device initialization	Executing the *RST command returns the Device-dependent functions to a known state that is independent of the past-use history of the device.

2.7.1 Bus initialization

IFC Bus initialization by the IFC statement

Function Keeps the IFC line active for approximately 100 μ s to initialize the interface functions of all devices connected to the GPIB bus line. Only the system controller can transfer the IFC command.

2.7.2 Message exchange initialization

DCL, SDC Initialization for message exchange by DCL and SDC bus command

Function The DCL and SDC command initialize the message exchange function for all devices on the GPIB or for specified devices only on the GPIB respectively.

The purpose of initialization for message exchange is to prepare devices to receive new commands from the controller when the sections related to the message exchange inside the devices are in a state unsuitable for control from the controller as a result of the execution of other programs, etc.

DCL: Initializes message exchange for all device connected to bus.

SDC: Initializes message exchange only for the device whose address is specified.

2.7.3 Device initialization

***RST** Device initialization by the *RST command

Function Returns the Device-dependent functions to a known state that is independent of the past-use history of the device. When this instrument is switched on, it returns to the state it was in before the power was switched off. (For device status at power-on, see paragraph 2.7.4)

2.7.4 Device status at power-on

When the power is switched on:

1. The device is set to the same status as when the power was last switched off.
2. The Input Buffer and the Output Queue are cleared.
3. The parser, Execution Control, and Response Formatter sections are reset.

SECTION 3

RS-232C INTERFACE

This section describes the setting methods and so forth when the RS-232C interface is used as an option.

TABLE OF CONTENTS

3.1	Device Message List	3-3
3.2	Installing RS-232C Interface	3-5
3.3	Connecting RS-232C Cable	3-6
3.4	Setting RS-232C	3-7
3.5	Device Initialization	3-9
3.5.1	Device initialization	3-9
3.5.2	Device status at power-on	3-9

(Blank)

SECTION 3 RS-232C INTERFACE

3.1 Device Message List

A device message is a data message that is transferred between a controller and a device via the system interface. Data messages consist of two types: program messages and response messages. A program message is an ASCII data message transferred from the controller to the device. A response message is a data message sent from the device to the controller.

Some types of program and response messages are shown below.

Table 3-1 Device Message List

Program message (For details see Section 4)	Response message (For details see Section 5)
Program command Device-dependent command (For details see Section 9) IEEE488.2 common command (for details see Section 6) Program query	Status message (For details see Section 7) Response message

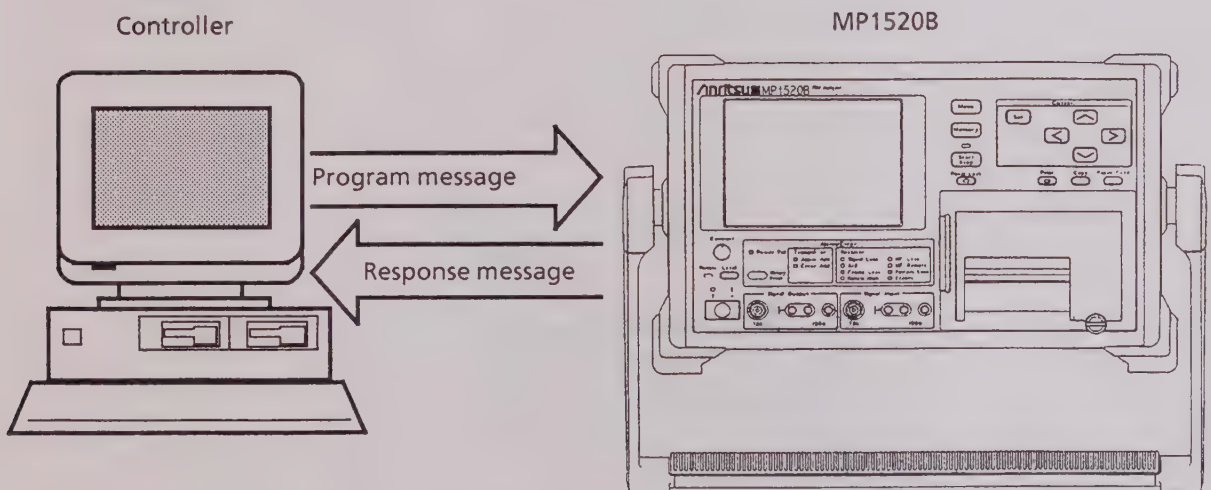


Fig. 3-1 Device Messages

The messages described previously are transferred via the input and output buffers of the device. The table below briefly describes the input and output buffers.

Table 3-2 Input and Output Buffers

Input buffer	Output buffer
A FIFO (First In First Out) memory area where DAB (program messages or query messages) are stored temporarily before they are parsed. The input buffer size of this instrument is 256 bytes.	A FIFO queue memory area. All DAB (response messages) output from a device to the controller are stored in this area until read by the controller.

3.2 Installing RS-232C Interface

Use the RS-232C interface (OPT05 Plug-in Unit) by plugging the Plug-in Unit into the slot as shown below

CAUTION

When replacing the plug-in unit, be careful not to injure your fingers.

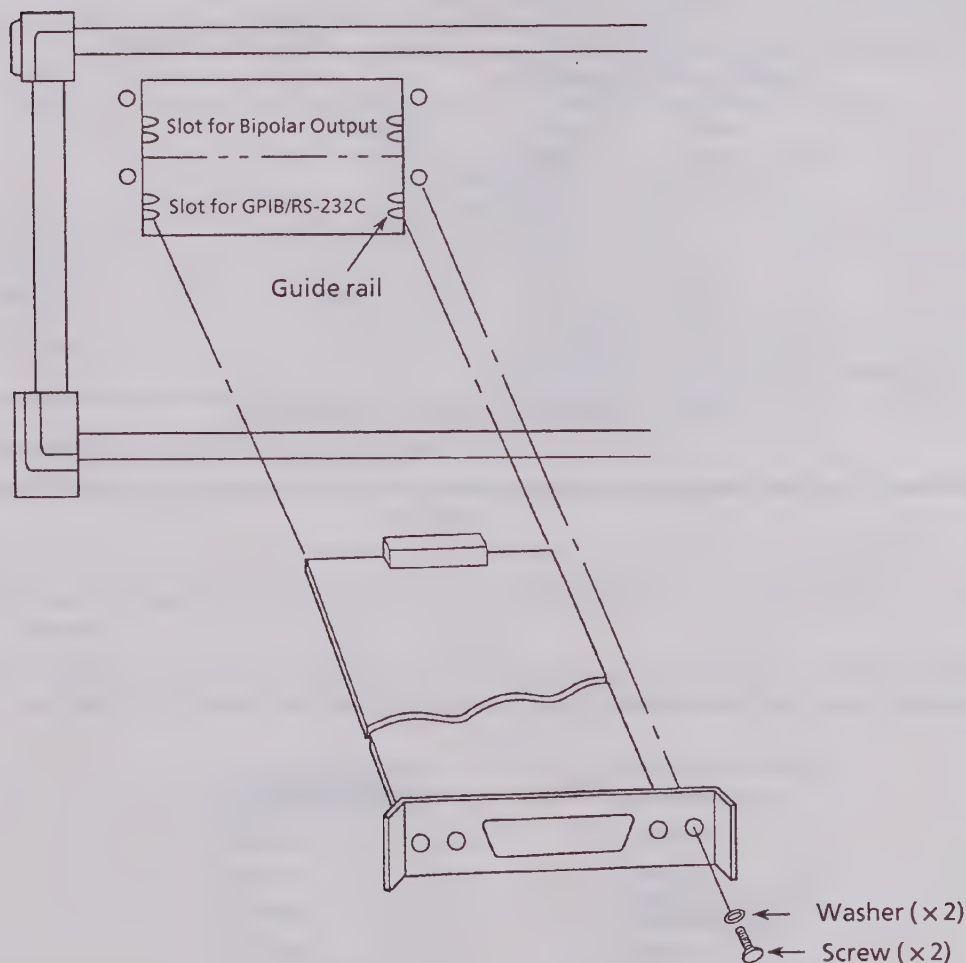


Fig. 3-2 Plugging in the Plug-in Unit

Step	Procedure
1	Check that the power switch has been turned OFF before replacing the plug-in unit.
2	Check which slot position to use in Fig. 2-3 before plugging in the plug-in unit.
3	To plug the plug-in unit into the slot, fit the unit end accurately to the two guide rails inside the slot, and insert the unit to the inner part so that its connector can be mated firmly, then tighten the right-to-left screws. Plugging in the unit or tightening the screws incorrectly may cause faulty operation.
4	Although a slot which does not contain a plug-in unit may not affect the operation of this instrument, it is recommended that such a slot be covered with a blank panel.

3.3 Connecting RS-232C Cable

Connect the RS-232C cable to the RS-232C connector on the MP1520B rear panel as shown in Fig. 3-3.

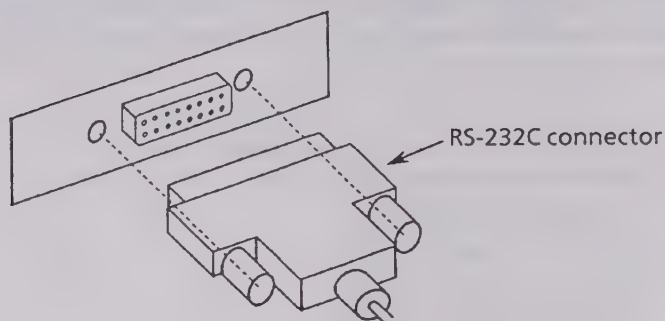


Fig. 3-3 RS-232C Cable Connection

3.4 Setting RS-232C

To use RS-232C as an automatic control interface, place the instrument in local mode and then make the settings shown below using the front-panel keys.

Table 3-3 Setting Contents in RS-232C Interface Use

Setting contents	Setting item	Setting range
How to use interface	Usage	Control: Used for remote control Printer: Used for external printer
Bit rate	Speed	19200bps, 9600bps, 4800bps, 2400bps, 1200bps, 600bps, 300bps
Parity	Parity	None: None Even: Even parity Odd: Odd parity
Character length	Character Length	8bits, 7bits
Stop bit length	Stop Bit Length	2bits, 1.5bits, 1bit
Flow control method	Flow Control	Ready/Busy: RS-232C control line is used XON/XOFF: XON/XOFF characters are used

Make the above settings using the System screen.

Step	Procedure
1	(1) Press the Menu key, and a window as shown below opens in the upper-right corner of the screen. (2) Use the Cursor key to position the window cursor on the item labeled System then press the Set key (or Menu key), and the current screen is switched to the System screen.

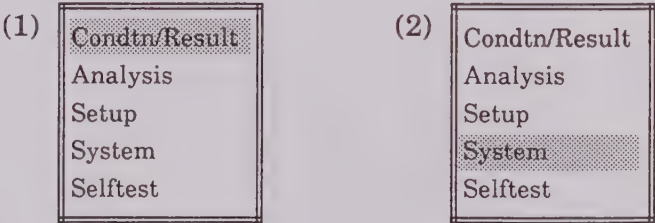


Fig. 3-4 Screen Selection Window

Step	Procedure
------	-----------

- 2 2 The operation at Step 1 causes transition to the System screen shown below. Move the reverse cursor to the *1 through *6 positions to make appropriate settings. For details of settings, refer to the panel operating instruction part of the separate Operation Manual.

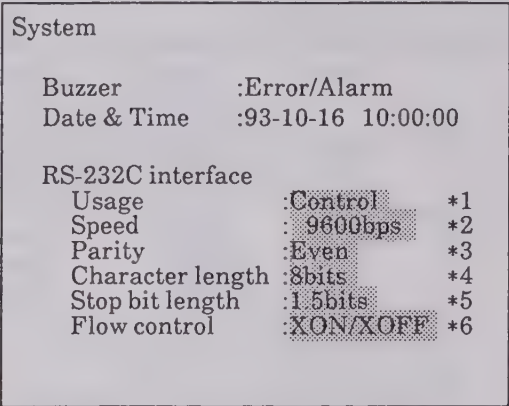


Fig. 3-5 System Screen (While RS-232C interface is being operated)

3.5 Device Initialization

The *RST command is supported to initialize a device when the RS-232C interface is used.

3.5.1 Device initialization

***RST** Device initialization by the *RST command

Function Returns the Device-dependent functions to a known state that is independent of the past-use history of the device.

When this instrument is switched on it returns to the state it was in before the power was switched off. (For device status at power-on, see paragraph 3.5.2)

3.5.2 Device status at power-on

When the power is switched on:

1. The device is set to the same status as when the power was last switched off.
2. The Input Buffer and the Output Queue are cleared.
3. The parser, Execution Control, and Response Formatter sections are reset.

(Blank)

SECTION 4

LISTENER INPUT FORMAT

This section describes the format of the program messages which the listener (this instrument) receives from the talker (controller).

Since the Device-dependent commands used in this instrument conform to SCPI, SCPI commands are used for explaining the examples in this section.

TABLE OF CONTENTS

4.1	Listener Input Program Message Format	4-3
4.2	Functional Elements of Program Messages	4-5
4.2.1	<TERMINATED PROGRAM MESSAGE>	4-5
4.2.2	<PROGRAM MESSAGE TERMINATOR>	4-6
4.2.3	<white space>	4-7
4.2.4	<PROGRAM MESSAGE>	4-7
4.2.5	<PROGRAM MESSAGE UNIT SEPARATOR>	4-8
4.2.6	<PROGRAM MESSAGE UNIT>	4-8
4.2.7	<COMMAND MESSAGE UNIT> and <QUERY MESSAGE UNIT>	4-9
4.2.8	<COMMAND PROGRAM HEADER>	4-10
4.2.9	<QUERY PROGRAM HEADER>	4-12
4.2.10	<PROGRAM HEADER SEPARATOR>	4-13
4.2.11	<PROGRAM DATA SEPARATOR>	4-13
4.3	Program Data Format	4-14
4.3.1	<CHARACTER PROGRAM DATA>	4-15
4.3.2	<DECIMAL NUMERIC PROGRAM DATA>	4-16
4.3.3	<BOOLEAN PROGRAM DATA>	4-17
4.3.4	<STRING PROGRAM DATA>	4-18

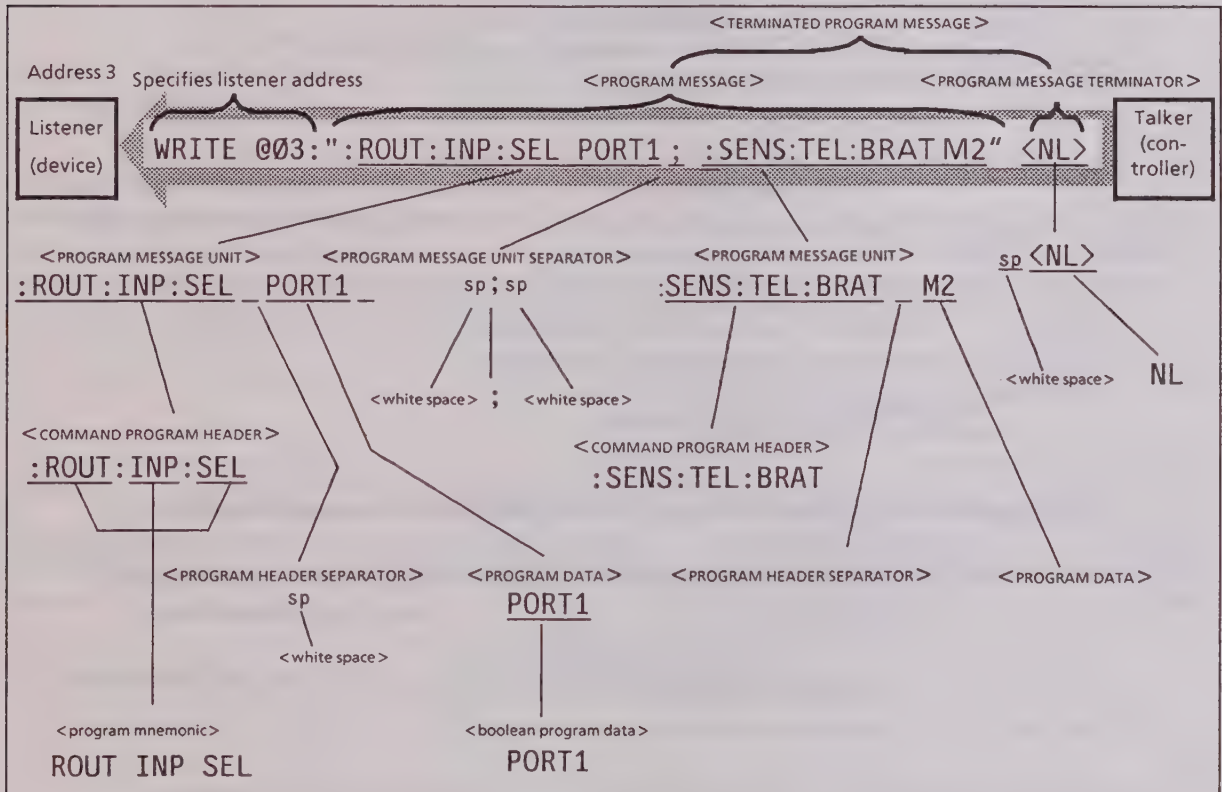
(Blank)

SECTION 4

LISTENER INPUT FORMAT

4.1 Listener Input Program Message Format

The program message example illustrated below shows the settings when an unbalanced connector is selected for the input connector to set the receiver signal to 2 M.



The program message format is composed of a sequence of functional elements which is the minimum unit capable of expressing the function. In the examples in the above diagram, the functional elements are written in upper-case alphabetic characters between brackets **<>**. When the functional elements can be subdivided further, those elements are called 'encoding elements'. In the examples in the above diagram, the encoding elements are written in lower-case alphabetic characters between brackets **<>**.

The program message formats are explained using a functional syntax diagram and an encoding syntax diagram on the following page.

- Functional syntax diagram: Indicates the selection of functional elements on a specific path
- Encoding syntax diagram: Indicates the selection of encoding elements on a specific path

The format of WRITE/READ commands is as follows.

WRITE @

Outputs data to the device.

★ Syntax

WRITE @device number: data

Data→

Numeric-Expression

string-expression

<Example>

Same as preceding example

WRITE @03:" :ROUT:INP:SEL PORT1 ; :SENS:TEL:BRAT M2"

Listener address (When 3 is set to the GPIB address of PDH analyzer)

READ @

Assigns data transferred from the device to a variable.

★ Syntax

READ @device number:variable

<Example>

Assign data transferred from the device (Input connector setting) to variable A\$.

WRITE @03:" :ROUT:INP:SEL?" ← Queries the input connector which has been selected.

READ @03:A\$

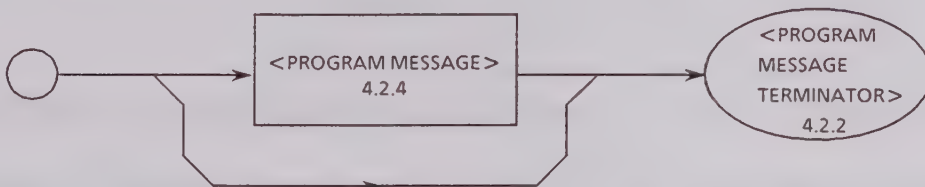
Listener address (Specifies the PDH Analyzer whose GPIB address is 3.)

4.2 Functional Elements of Program Messages

This measuring instrument accepts a program message by detecting the terminator at the end of it. The functional elements of program messages are explained below.

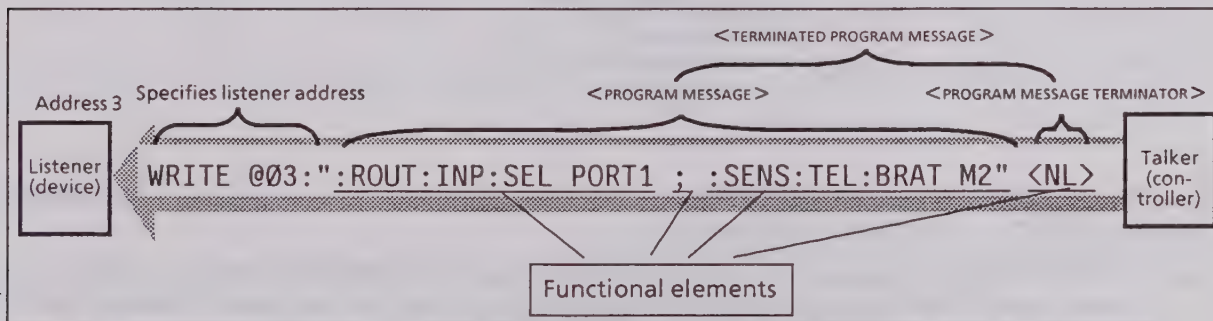
4.2.1 <TERMINATED PROGRAM MESSAGE>

A <TERMINATED PROGRAM MESSAGE> is defined as follows.



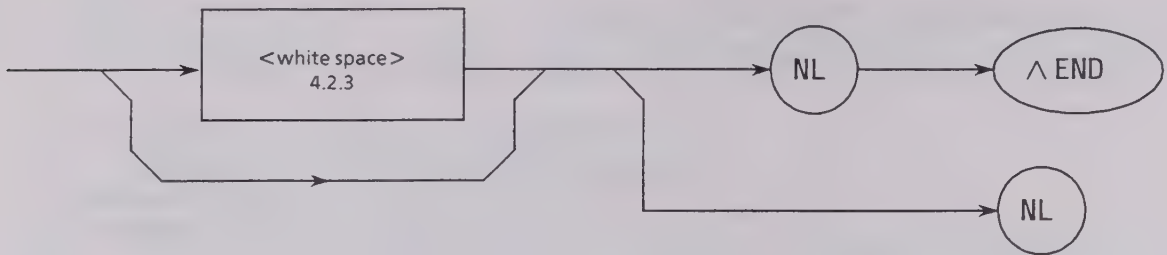
A <TERMINATED PROGRAM MESSAGE> is a data message which has all the functional elements required for transmission from the controller to this measuring instrument. A <PROGRAM MESSAGE TERMINATOR> is attached to the end of a <PROGRAM MESSAGE> to terminate its transmission.

<Example> <TERMINATED PROGRAM MESSAGE> which sends 2 commands .

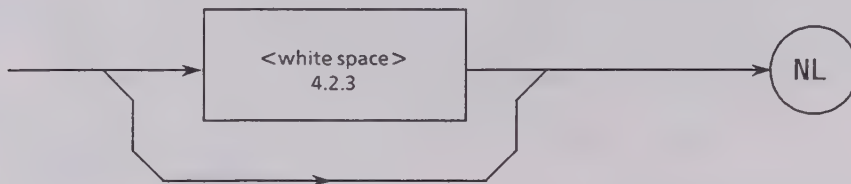


4.2.2 <PROGRAM MESSAGE TERMINATOR>

(1) A <PROGRAM MESSAGE TERMINATOR> used for GPIB interface is defined as follows.



(2) A <PROGRAM MESSAGE TERMINATOR> used for RS-232C interface is defined as follows.



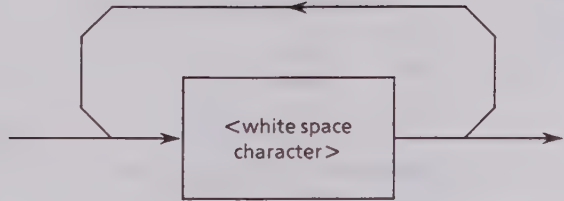
A <PROGRAM MESSAGE TERMINATOR> terminates a sequence of one or more <PROGRAM MESSAGE UNIT> elements of a fixed length.

NL: NL is defined as a single ASCII code byte 0A (decimal 10), i.e. the ASCII control code LF (Line Feed) used to return the carriage and bring the print position to the beginning of the next line. It is also called NL (New Line).

END: The EOI signal can be generated by making the EOI line (one of GP-IB management bus lines) true (low level). EOI ON/OFF is one statement for controlling the EOI line.

4.2.3 <white space>

A <white space> is defined as follows.

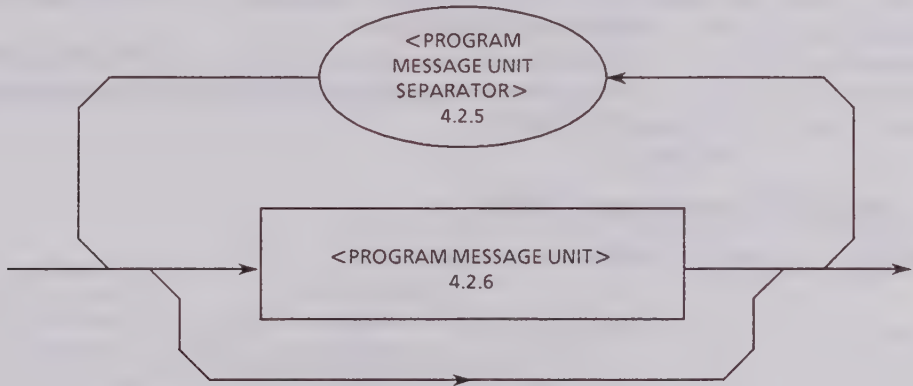


A <white space> is defined as a single ASCII code byte in the range 00 to 09, 0B to 20 (decimal 0 to 9, 11 to 32).

This range includes ASCII control signals and space signal except new line. This measuring instrument either treats them as ASCII control signals but as spaces, or skips over them.

4.2.4 <PROGRAM MESSAGE>

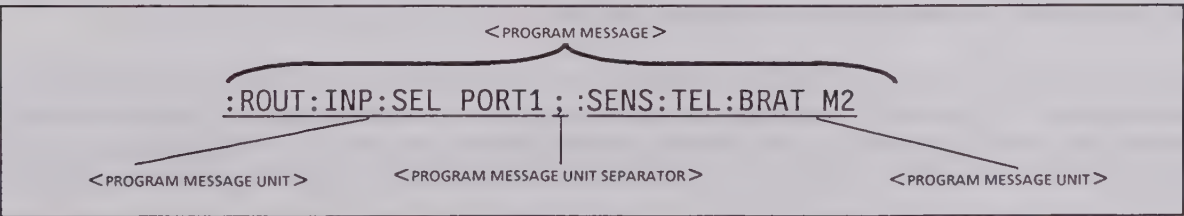
A <PROGRAM MESSAGE> is defined as follows.



A <PROGRAM MESSAGE> consists of zeros, one or several <PROGRAM MESSAGE UNIT> elements. <PROGRAM MESSAGE UNIT> elements are either programming commands or data sent from the controller to this measuring instrument. The <PROGRAM MESSAGE UNIT SEPARATOR> element is used to separate <PROGRAM MESSAGE UNITS>.

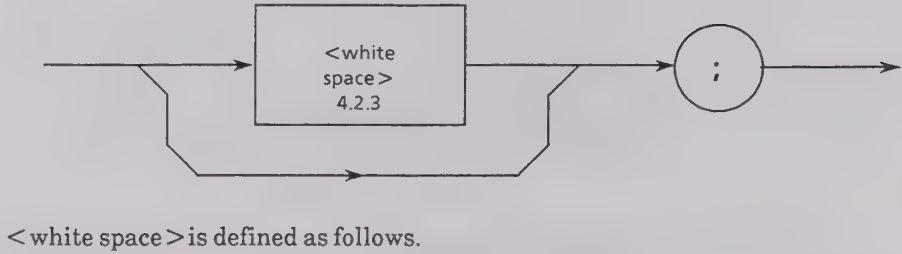
<Example>

Select the unbalanced connector and set the receiver signal to 2 M.

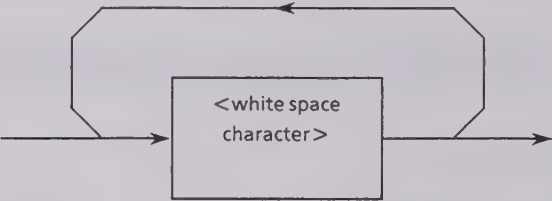


4.2.5 <PROGRAM MESSAGE UNIT SEPARATOR>

A <PROGRAM MESSAGE UNIT SEPARATOR> is defined as follows.



<white space> is defined as follows.

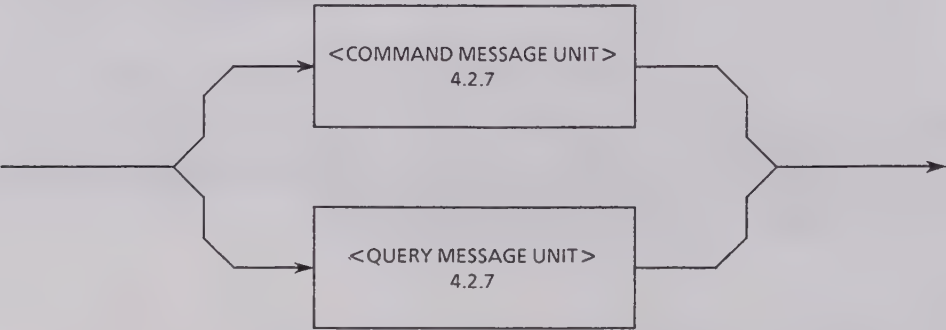


The <PROGRAM MESSAGE UNIT SEPARATOR> separates the <PROGRAM MESSAGE UNIT> elements in the range of a <PROGRAM MESSAGE>.

This measuring instrument interprets a semicolon as the separator of <PROGRAM MESSAGE UNIT> elements so, it skips the <white space characters> before and after the semicolon. <white space characters> make a program easy to read.

4.2.6 <PROGRAM MESSAGE UNIT>

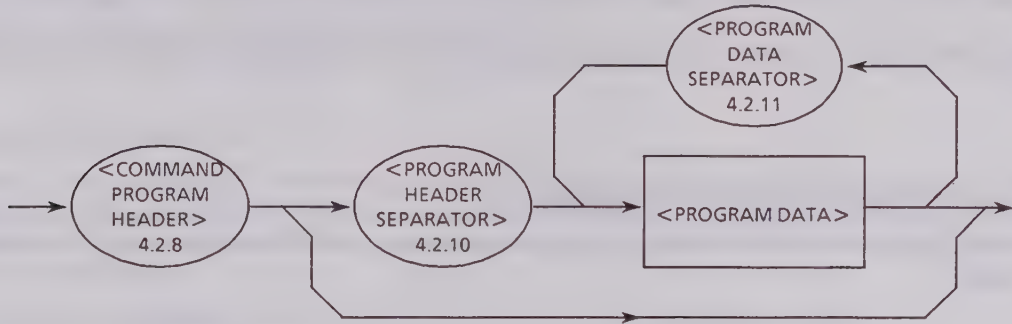
A <PROGRAM MESSAGE UNIT> is defined as follows.



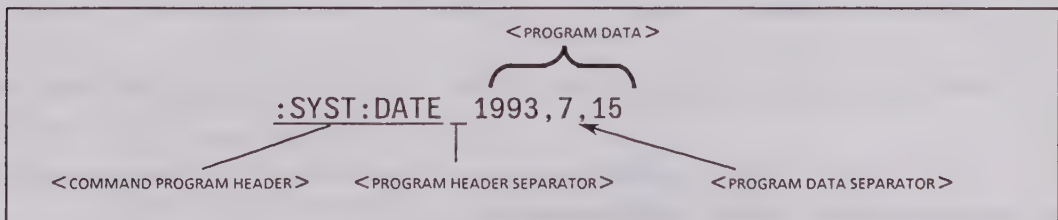
A <PROGRAM MESSAGE UNIT> is either the <COMMAND MESSAGE UNIT> or <QUERY MESSAGE UNIT> received by this measuring instrument.

4.2.7 <COMMAND MESSAGE UNIT> and <QUERY MESSAGE UNIT>

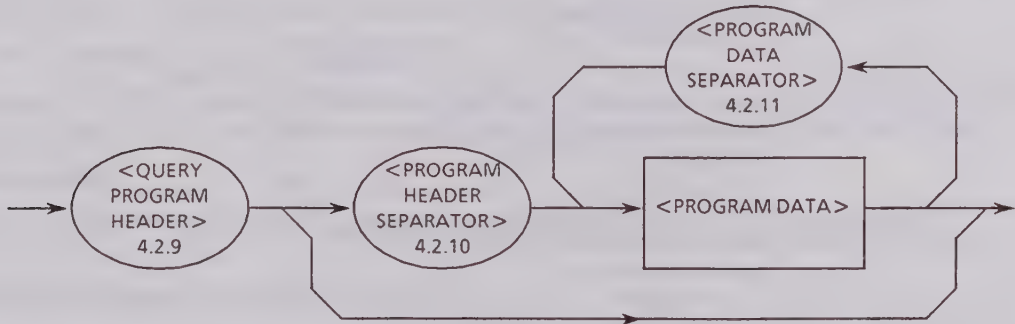
(1) A <COMMAND MESSAGE UNIT> is defined as follows.



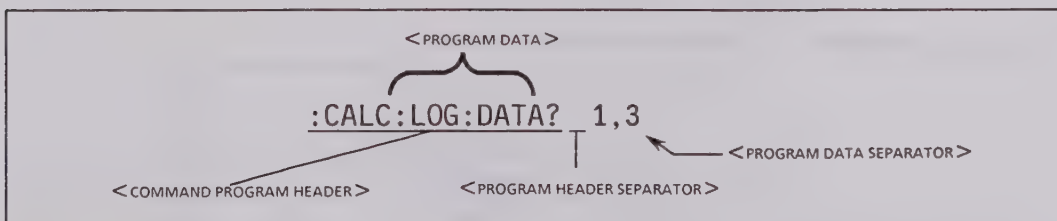
<Example>: Set the date.



(2) A <QUERY MESSAGE UNIT> is defined as follows.



<Example>: Query the log data.

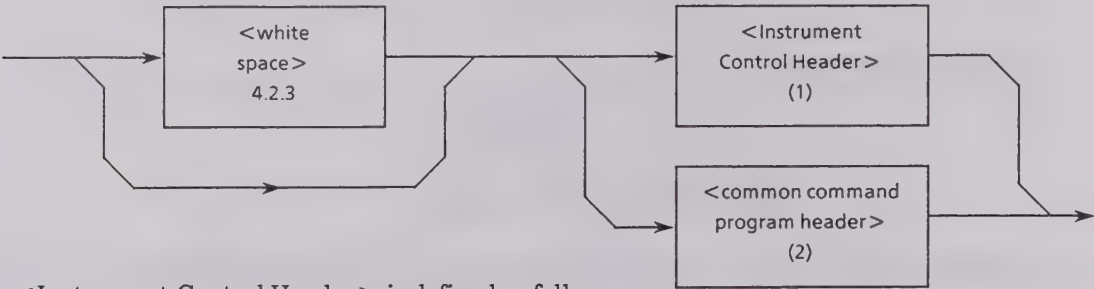


For both <COMMAND MESSAGE UNITS> and <QUERY MESSAGE UNITS>, a space must be inserted between the program header and any program data immediately following it. The application, function and operation of the program data can be seen from the program header. If there is no program data; the application, function or operation to be performed by a device is indicated by the header alone.

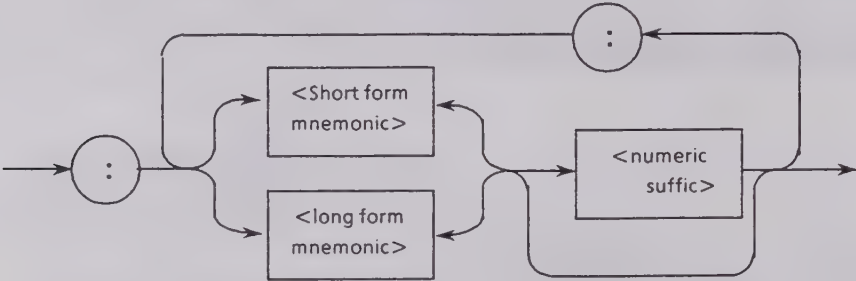
The <COMMAND PROGRAM HEADER> is a command by which the controller controls this measuring instrument. The <QUERY PROGRAM HEADER> is a command used for sending a query from the controller to this measuring instrument so that the controller can receive a response message from it.
The special feature of the header is that a question mark is always tagged on at the end to indicate that it is a query.

4.2.8 <COMMAND PROGRAM HEADER>

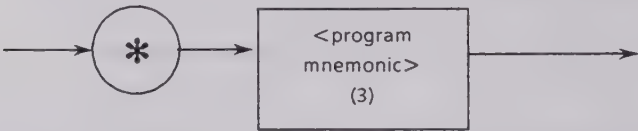
A <COMMAND PROGRAM HEADER> is defined as follows. A <white space> may be placed in front of each header.



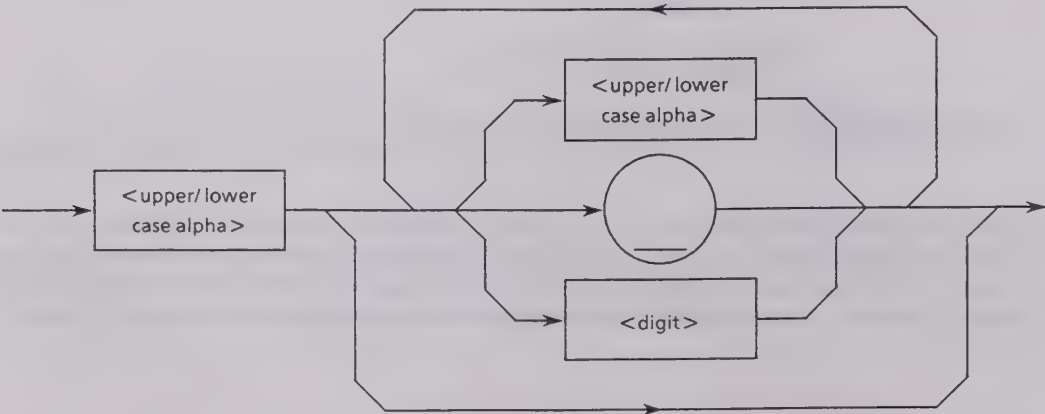
(1) A <Instrument-Control Header> is defined as follows.



(2) A <common command program header> is defined as follows.



(3) A <program mnemonic> is defined as follows.



■ <COMMAND PROGRAM HEADER>

Indicates the application, function and operation of a program to be executed by this measuring instrument. If there is no program data; the application, function and operation to be executed by this measuring instrument are indicated in the header itself.

This is expressed in ASCII code characters by a <program mnemonic>, usually called just mnemonic.

■ <program mnemonic>

A mnemonic must begin with upper-case or lower-case alphabetic characters. Following that, upper-case alphabetic characters from A to Z, lower-case alphabetic characters, the underline and numbers from 1 to 9 can be used in any combination. The maximum length of a mnemonic is 12 characters. There are no spaces between characters.

● <upper/lower-case alpha>

Defined as a single ASCII code byte in the range 41 to 5A, 61 to 7A (decimal 65 to 90, 97 to 122 = A to Z, a to z).

● <digit>

Digits are defined as single ASCII code byte in the range 30 to 39 (decimal 48 to 57 = numeric 0 to 9).

● (_)

The underline is defined as the single ASCII code byte 5F (decimal 95).

■ <Instrument -control Headers >

An <Instrument-Control Header> is defined in SCPI. Since the Device-dependent commands used for this instrument conform to SCPI, the command format is expressed as in SCPI.

● <short form mnemonic>/<long form mnemonic>

Corresponds to the SCPI command short form and long form respectively. (For details see Section 8). The mnemonic definition is identical to the <program mnemonic> definition explained above.

● <numeric suffix>

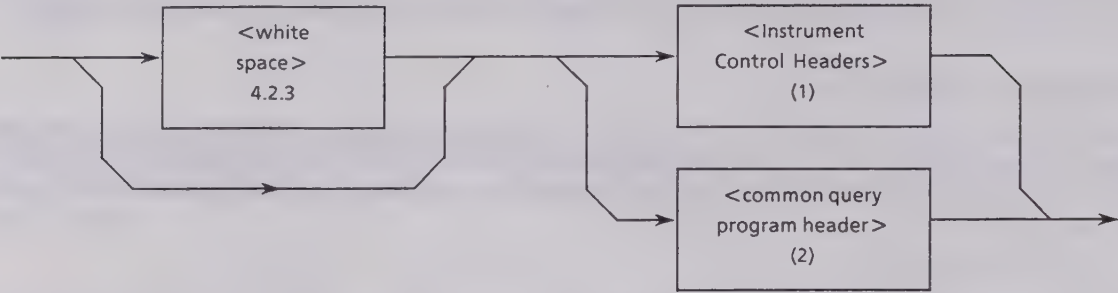
Defined as single ASCII code byte in the range 30 to 39 (decimal 48 to 57 = numeric 0 to 9).

■ <common command program header>

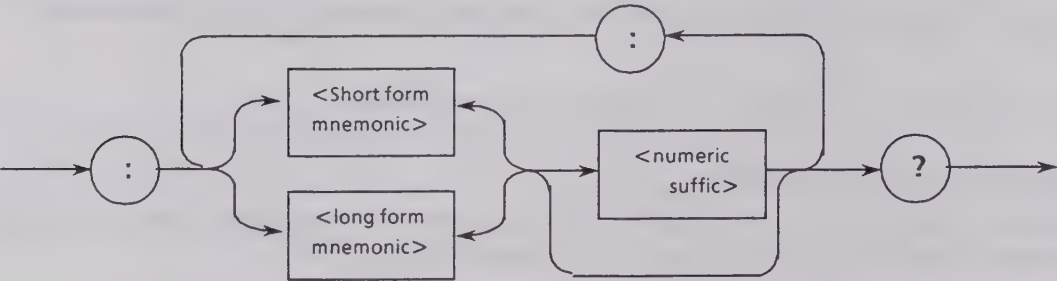
An asterisk is always placed before the <program mnemonic> in a <common command program header>.

4.2.9 <QUERY PROGRAM HEADER>

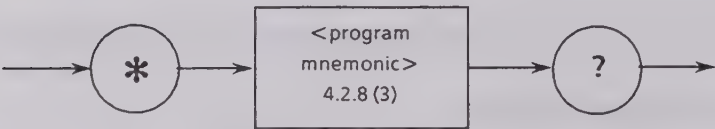
A <QUERY PROGRAM HEADER> is defined as follows.
A <white space> is placed before each header.



(1) A <Instrument-Control Header> is defined as follows.



(2) A <common query program header> is defined as follows.



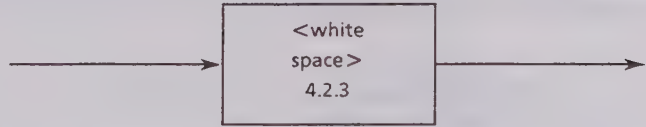
■ <QUERY PROGRAM HEADER>

A <QUERY PROGRAM HEADER> is a command for sending a query from the controller to this measuring instrument so that the controller can receive a response message from it. A ? is always added at the end of the header to indicate a query.

Except for the ? after it, the format of the <QUERY PROGRAM HEADER> is identical to that of the <COMMAND PROGRAM HEADER> described in paragraph 4.2.8.

4.2.10 <PROGRAM HEADER SEPARATOR>

A <PROGRAM HEADER SEPARATOR> is defined as follows.



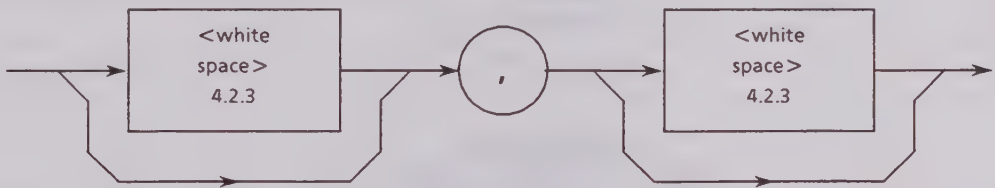
A <PROGRAM HEADER SEPARATOR> is used to separate a <COMMAND PROGRAM HEADER> or <QUERY PROGRAM HEADER> from <PROGRAM DATA>.

When there is more than one <white space character> between a program header and program data, the first is interpreted as the separator and the rest are skipped. <white space characters> are used to make a program easy to read.

So, there must always be one header separator between the header and the data to indicate the end of the program header and the start of the program data.

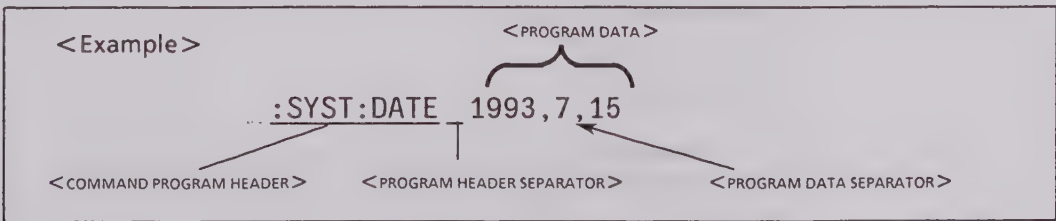
4.2.11 <PROGRAM DATA SEPARATOR>

A <PROGRAM DATA SEPARATOR> is defined as follows.



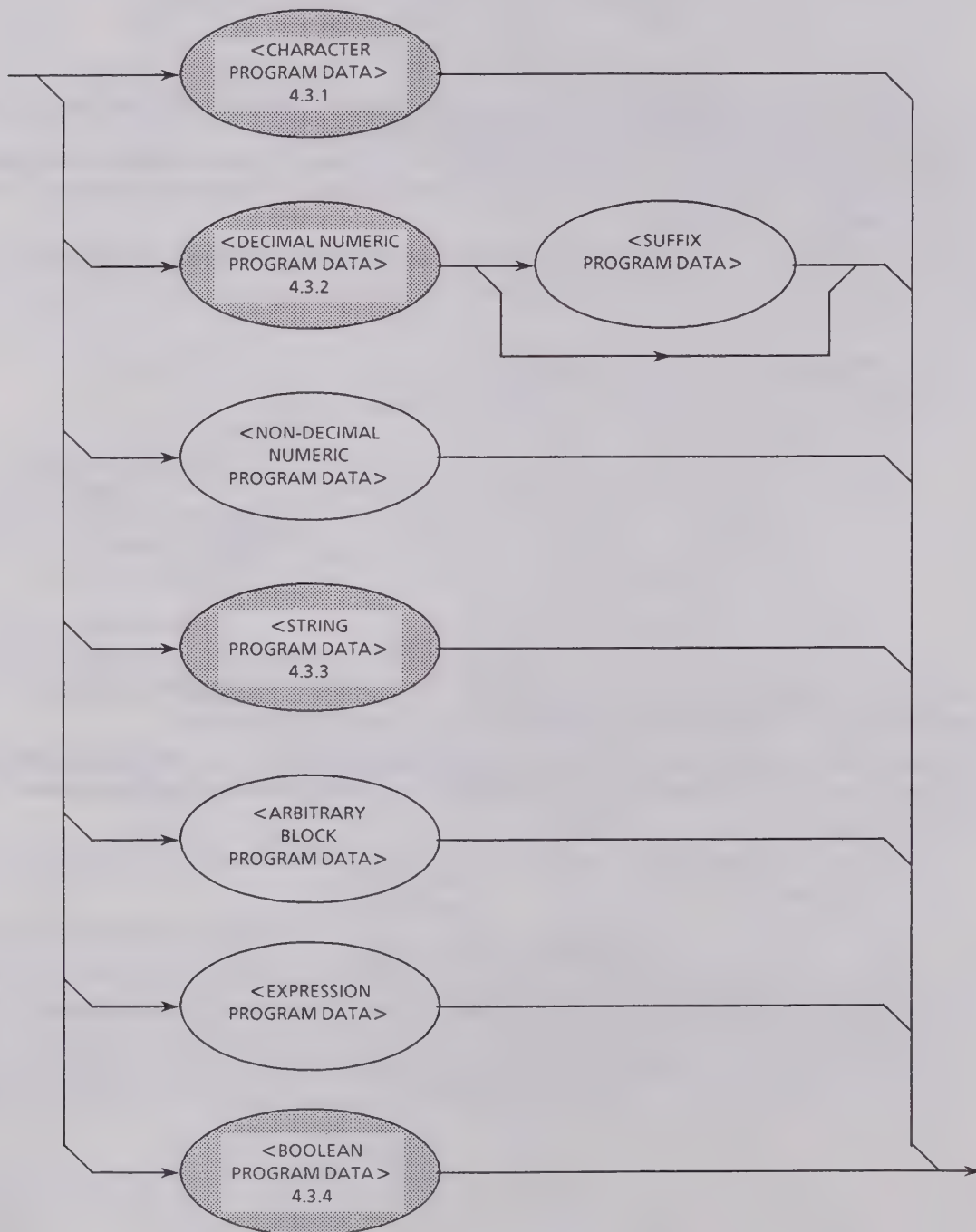
When a <COMMAND PROGRAM HEADER> or <QUERY PROGRAM HEADER> has many parameters, A <PROGRAM DATA SEPARATOR> is used to separate them.

A comma must be used with a <PROGRAM DATA SEPARATOR>, but a white space does not always have to be used. A white space before or after the comma is skipped. They are used to make a program easier to read.



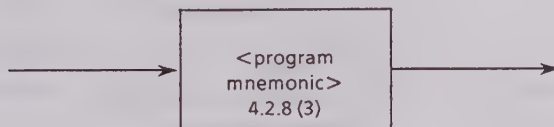
4.3 Program Data Format

The following describes the format of <PROGRAM DATA> in the functional syntax diagram on paragraph 4.2.7. <PROGRAM DATA> functional elements are used in sending various types of parameter related to the program header. The diagram below shows the different types of program data..



4.3.1 <CHARACTER PROGRAM DATA>

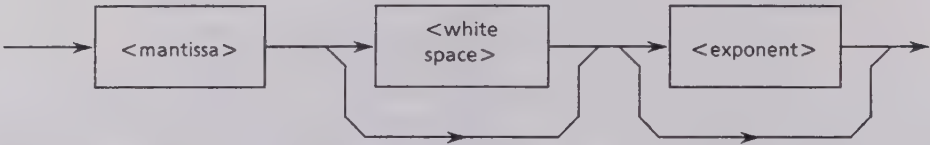
A <CHARACTER PROGRAM DATA> item, which indicates a short mnemonic, is defined as follows.



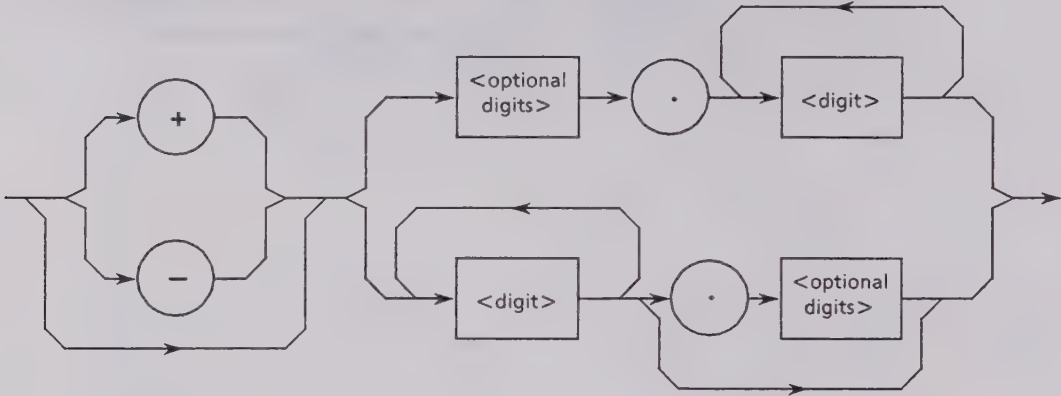
<Example> :SENSe:TELEcom:PATtern:TYPE PRBS11 (Mnemonic data indicating PRBS²¹¹ – 1)
:SENSe:MEASure:TYPE SINGLE (Mnemonic data indicating single measurement)
:calcULATE:LOG:formATEC (Mnemonic data indicating error count)

4.3.2 <DECIMAL NUMERIC PROGRAM DATA>

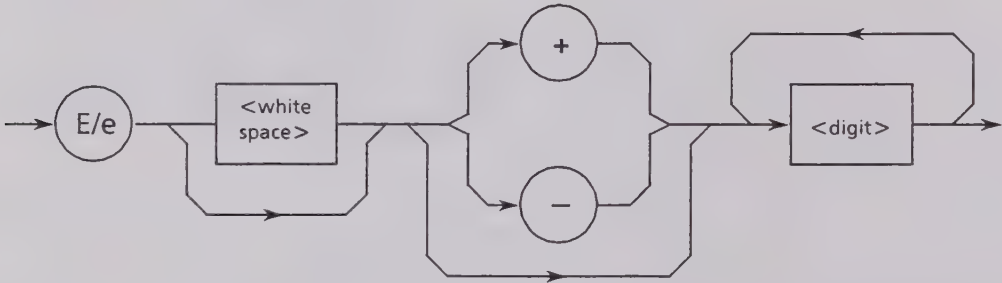
A <DECIMAL NUMERIC PROGRAM DATA> item, which indicates a decimal integer, is defined as follows.



A <mantissa> is defined as follows:



An <exponent> is defined as follows:



This instrument employs the integer format and real number format expressed in decimal notation.

■ Integer format

Indicates an integer value expressed in decimal notation.

- Zeros can be inserted at the beginning. 005
- Spaces cannot be inserted between a sign and a number. +5 (○), +△5(×)
- Spaces can be inserted after a number. +5△△
- A + sign can be optionally appended. +5, 5
- Commas cannot be used to separate digits 1,234(×)

<Example> :SOURce:TELEcom:OFFSet 5

■ Real number format

Indicates a real number expressed in decimal notation. This format, which is expressed with or without an exponent part, can be used only for setting the threshold value for performance.

- Spaces cannot be inserted between a number and a decimal point +123△.456 (×)
- Spaces can be inserted after the fraction part and in the exponent part 123.456△ E△ +78 (○)
- There need not be any numbers before the decimal point. .05 (○)

<Example> :CALC:TEL;PERF;THR:G821:ES:S1 12.3456
:CALC:TEL:PERF:THR:G826:ESR:S1 1.00E-08

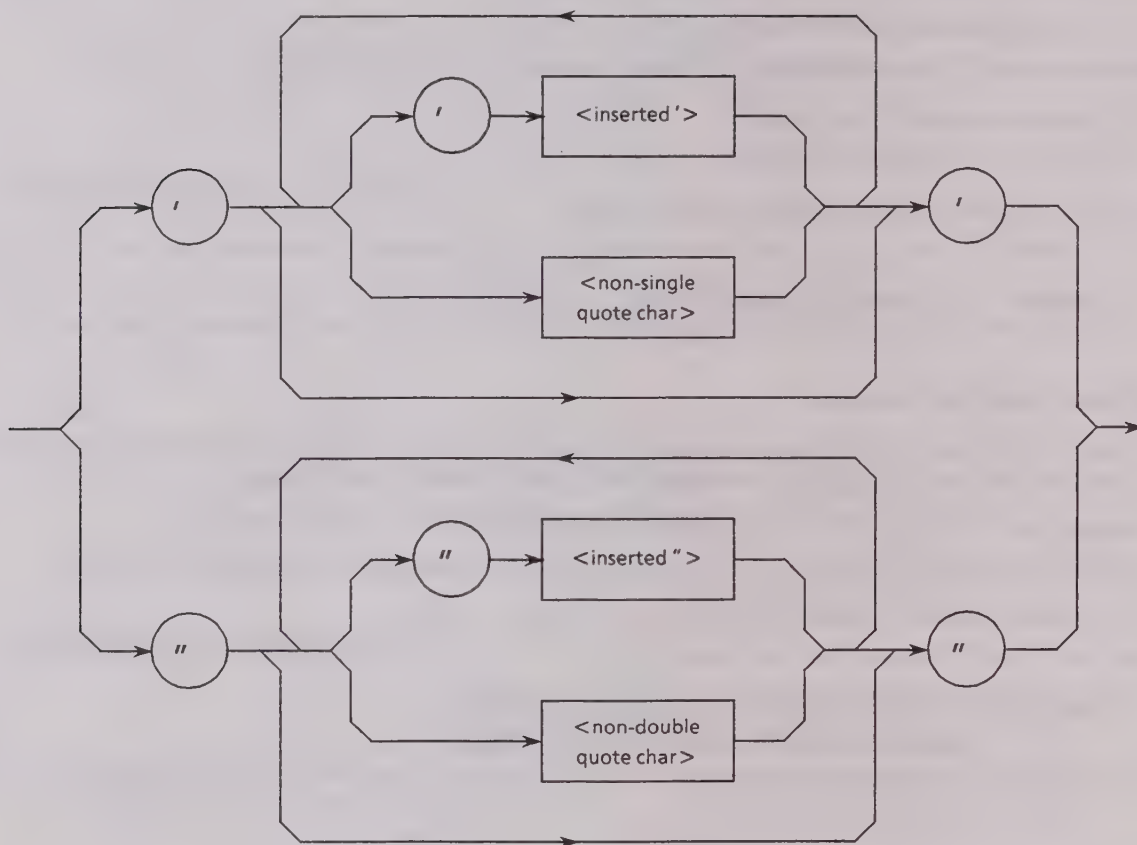
4.3.3 <BOOLEAN PROGRAM DATA>

A <BOOLEAN PROGRAM DATA> item, which is defined by SCPI, represents a theoretical value. The TRUE and FALSE values are defined so that the ON and OFF of a <CHARACTER PROGRAM DATA> item correspond to the 1 and 0 respectively of a <DECIMAL NUMERIC PROGRAM DATA> item.

<Example> :SOURce:TELEcom:FRAMing ON
:SOURce:TELEcom:FRAMing 1

4.3.4 <STRING PROGRAM DATA>

A <STRING PROGRAM DATA> item is a character string which has a single or double quotation mark at each end. It is defined as follows.



- (1) <inserted'> is defined as a single ASCII code byte of 27 (decimal 39 = ')
- (2) <non-single quote char> is defined as any single ASCII code byte other than 27 (decimal, 39 = ')
- (3) <inserted'>d is defined as a single ASCII code byte of 22 (decimal 34 = ")
- (4) <non-double quote char> is defined as any single ASCII code byte other than 22 (decimal, 34 = ")

<Example> :SENSe:TEL:PATT:UWORD "1100" (Character string indicating word data 1100)
 :DISP:MENUn:NAME "CONDtn:Tx" (Character string indicating Condtn TX screen)
 :CALC:DATA? "EC:BIT" (Character string indicating error count of bit error)

SECTION 5

TALKER OUTPUT FORMAT

This section describes the format of response messages sent to the listener (controller) from the talker (this instrument).

TABLE OF CONTENTS

5.1	Talker Output Response Message Format	5-3
5.2	Functional Elements of Response Message	5-4
5.2.1	<TERMINATED RESPONSE MESSAGE>	5-4
5.2.2	<RESPONSE MESSAGE TERMINATOR>	5-5
5.2.3	<RESPONSE MESSAGE>	5-6
5.2.4	<RESPONSE MESSAGE UNIT SEPARATOR>	5-6
5.2.5	<RESPONSE MESSAGE UNIT>	5-7
5.2.6	<RESPONSE DATA SEPARATOR>	5-7
5.2.7	<RESPONSE DATA	5-8

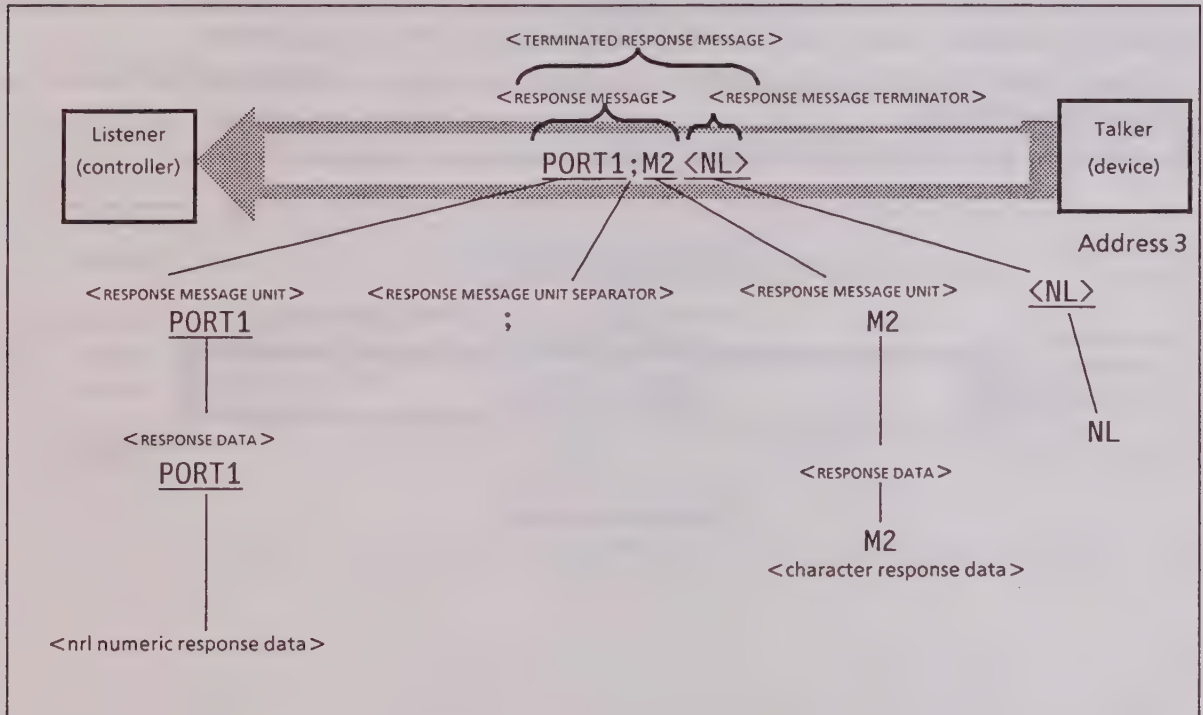
(Blank)

SECTION 5

TALKER OUTPUT FORMAT

5.1 Talker Output Response Message Format

The following illustrates the response messages sent to the controller from the talker after having transferred the :ROUT:INP:SEL? query command for querying the selected input connector and the :SENSE:TEL:BRAT? query command for querying the bit rate of the received signal. Since the SCPI response data used is headerless, only the data is returned.



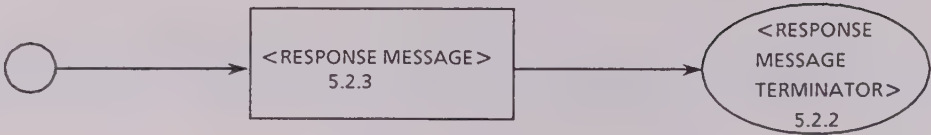
A response message format is composed of a sequence of functional elements that is the minimum unit capable of expressing the function.

In the above diagram, the functional elements are written in upper-case alphabetic characters between brackets <>. When the functional elements can be subdivided further, those are called encoding elements. In the examples in the diagram above, the encoding elements are written in lower-case alphabetic characters between brackets <>. Therefore, both the talker and listener formats are written in similar syntax notation.

5.2 Functional Elements of Response Message

5.2.1 <TERMINATED RESPONSE MESSAGE>

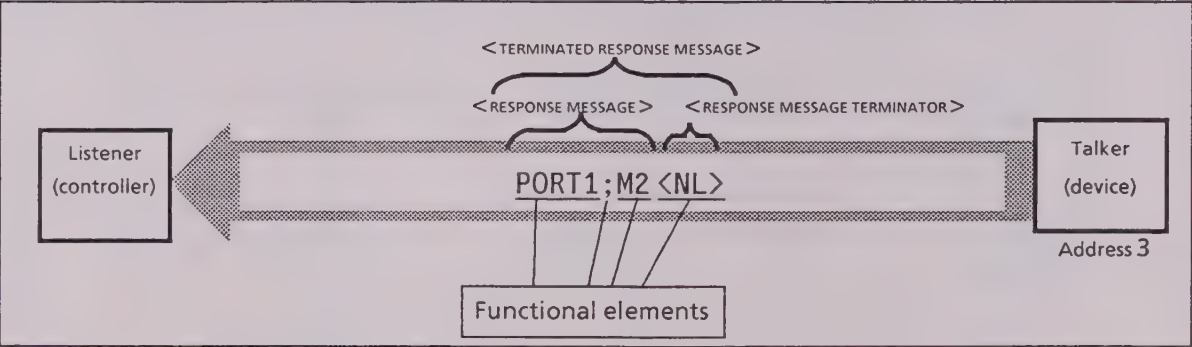
A <TERMINATED RESPONSE MESSAGE> is defined as follows:



A <TERMINATED RESPONSE MESSAGE> is a data message, containing all the functional elements required for transmission, sent from a talker this measuring instrument to the controller.

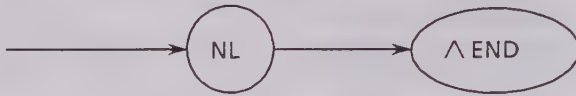
A <RESPONSE MESSAGE TERMINATOR> is attached to the end of a <RESPONSE MESSAGE> to terminate its transmission.

<Example> A <TERMINATED RESPONSE MESSAGE> comprising 2 message units



5.2.2 <RESPONSE MESSAGE TERMINATOR>

(1) A <RESPONSE MESSAGE TERMINATOR> used for GPIB interface is defined as follows.



(2) A <RESPONSE MESSAGE TERMINATOR> used for RS-232C interface is defined as follows.



A <RESPONSE MESSAGE TERMINATOR> is placed after the last <RESPONSE MESSAGE UNIT> to terminate a fixed length sequence consisting of one or more <RESPONSE MESSAGE UNIT> elements.

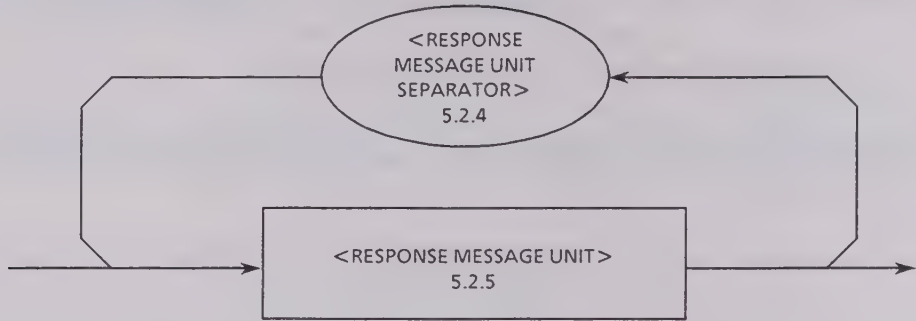
<Example> Program for reading out the input connector state which has been set.

```

10 WRITE @03:":ROUT:INP:SEL?"
20 READ @03:A$
30 PRINT A$
40 END
  
```

5.2.3 <RESPONSE MESSAGE>

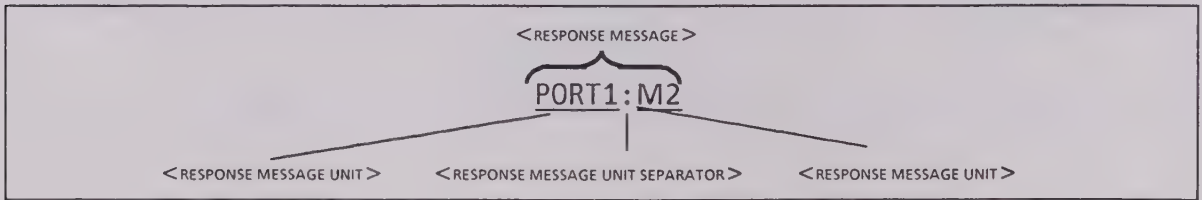
A <RESPONSE MESSAGE> is defined as follows.



A <RESPONSE MESSAGE> consists of one <RESPONSE MESSAGE UNIT> element or a sequence of many <RESPONSE MESSAGE UNIT> elements.

A <RESPONSE MESSAGE UNIT> element is a single message sent from this measuring instrument to the controller. A <RESPONSE MESSAGE UNIT SEPARATOR> element is used to separate <RESPONSE MESSAGE UNIT> elements.

<Example> Response messages when the selected input connector and the bit rate of the received signal were queried.



5.2.4 <RESPONSE MESSAGE UNIT SEPARATOR>

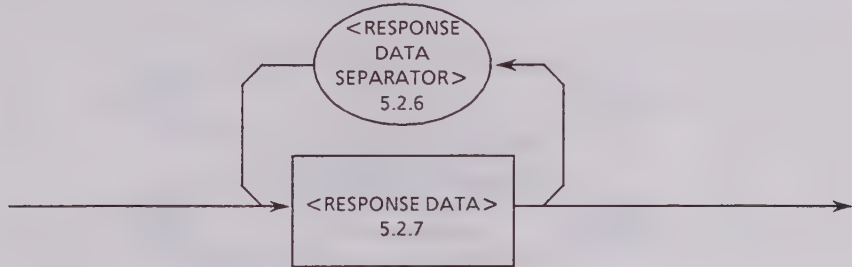
A <RESPONSE MESSAGE UNIT SEPARATOR> is defined as follows.



A semicolon (;) is used as the <RESPONSE MESSAGE SEPARATOR> to separate a sequence of <RESPONSE MESSAGE UNIT> elements that is to be transmitted as one message.

5.2.5 <RESPONSE MESSAGE UNIT>

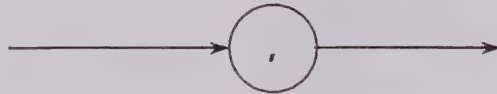
1. A <RESPONSE MESSAGE UNIT> is defined as follows.



2. The <RESPONSE MESSAGE UNIT> from this measuring instrument is a headerless response message unit and returns only the measured result data.

5.2.6 <RESPONSE DATA SEPARATOR>

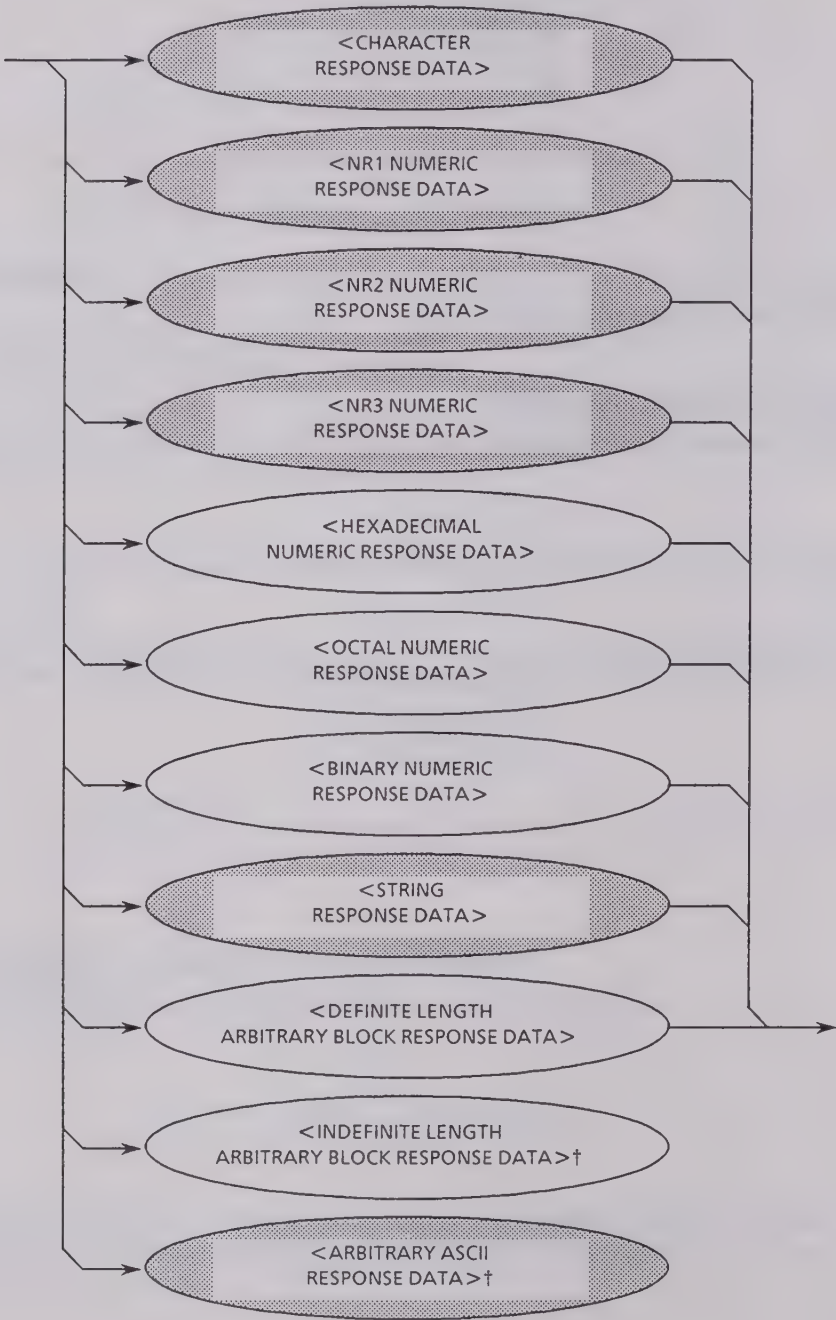
A <RESPONSE DATA SEPARATOR> is defined as follows:

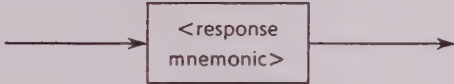
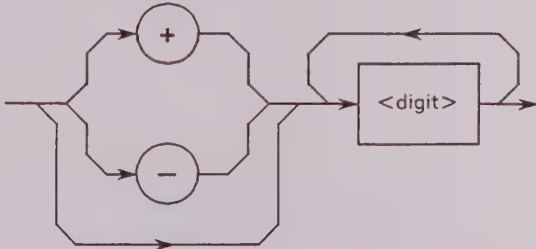
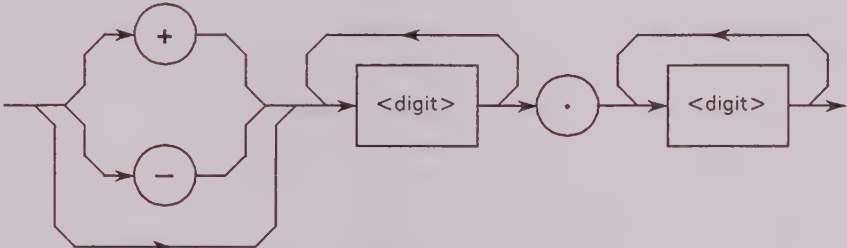
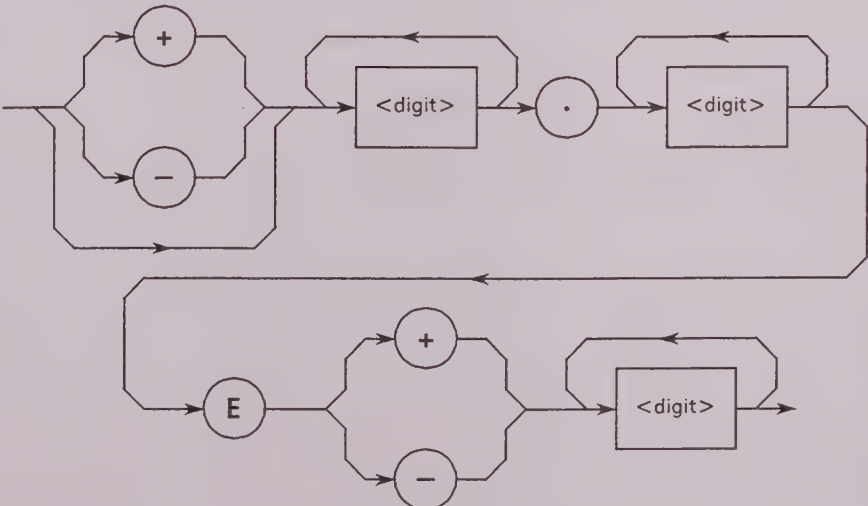


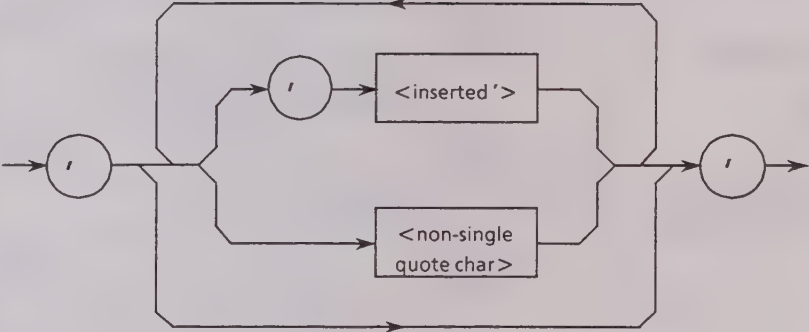
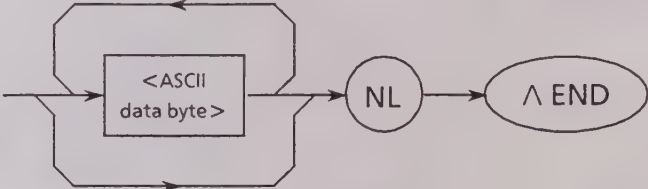
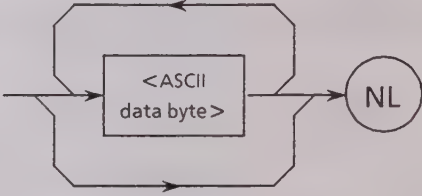
A <RESPONSE DATA SEPARATOR> is used to separate <RESPONSE DATA> items when more than one is output.

5.2.7 <RESPONSE DATA>

The diagram below shows the types of <RESPONSE DATA> used for this instrument. The type of response data to be returned is determined by the query message.



Element	Function
(1) CHARACTER RESPONSE DATA <Example> ABC DEFG	Indicates a short mnemonic data item. 
(2) NR1 NUMERIC RESPONSE DATA <Example> 123 +123 -1234	Indicates a decimal integer. 
(3) NR2 NUMERIC RESPONSE DATA <Example> 12.3 +12.34 -12.345	Indicates a fixed-point number. 
(4) NR3 NUMERIC RESPONSE DATA <Example> 1.23E+45 -12.3E+45	Indicates a decimal real number with exponent part. 

Element	Function
(5) STRING RESPONSE DATA <Example> "1234" "ABCD" "1234.5"	Indicates a character string whose both ends are enclosed with a double quotation mark. 
(6) ARBITRARY ASCII RESPONSE DATA <Example > <ASCII Byte><ASCII Byte>NL^END	ASCII data bytes (excluding NL characters) sent without separating them; so, the last data is terminated by NL^END. 1) When GPIB interface is used  2) When GPIB interface is used 

SECTION 6

IEEE488.2 COMMON COMMANDS

This section describes the IEEE488.2 common commands supported by this instrument. The common commands apply to both the GPIB and RS-232C interfaces. All the common commands supported by this instrument are sequential commands.

(Blank)

SECTION 6

IEEE488.2 COMMON COMMANDS

The table below lists the IEEE488.2 common commands supported by this instrument.

Table 6-1 IEEE488.2 Common Commands

Mnemonic	Full spelling
*IDN?	Identification Query
*RST	Reset Command
*TST?	Self Test Query
*OPC	Operation Complete Command
*OPC?	Operation Complete Query
*WAI	Wait Continue Command
*CLS	Clear Status Command
*ESE	Standard Event Status Enable Command
*ESE?	Standard Event Status Enable Query
*ESR?	Standard Event Status Register Query
*SRE	Service Request Enable Command
*SRE?	Service Request Enable Query
*STB?	Read Status Byte Query
*TRG	Trigger Command
*PSC	Power On Status Clear Command
*PSC?	Power On Status Clear Query
*SAV	Save Command
*RCL	Recall Command
*OPT?	Option Identification Query

■ *IDN? Identification Query

- Response <ARBITRARY ASCII RESPONSE DATA>
 <Manufacture's name>, <Model name>, <Serial number>,
 <Firmware revision number>
 Response data transferred by this instrument: ANRITSU,MP1520B,0,0
- Function Returns manufacture's name, model name etc. of product.
- Example *IDN?

■ *RST Reset Command

- Parameter None
- Function When the instrument is switched on, it returns to the state it was in before the power was switched off. At this time, the measured contents are cleared, but the measurement mode continues as it is. To return to the factory setting when shipped, set the :SYSTem:MEMory:RECall SCPI command parameter to <0>. Where <0>: Initial.
- Example *RST

■ *TST? Self Test Query

- Response <NR1 NUMERIC RESPONSE DATA>
 0: Without errors detected
 1: With errors detected
- Function Queries the results of the self-tests (RAM, LED, buzzer, etc. checks) executed at power-on. The self-tests (self-test executed on the Self-test screen) for signal, offset, and others are executed by the :TEST:START SCPI command. The test results are queried by the :TEST:DATA? query command.
- Example *TST?

■ *OPC Operation Complete Command

- Parameter None
- Function Sets the bit 0 (operation complete bit) in the Standard Event Status Register to set SRQ to ON when the previous execution has been completed.
- Example *OPC

■ *OPC? Operation Complete Query

- Response <NR1 NUMERIC RESPONSE DATA>
 1
- Function Returns 1 when the previous execution has been completed.
- Example *OPC?

■ *WAI Wait to Continue Command

- Parameter None
- Function Prevents this instrument from executing any further next commands until the previous execution has been completed.
- Example *WAI

■ *CLS Clear Status Command

- Parameter None
- Function Clears all event registers and queues except for the Output Queue and its MAV summary messages.

To reset the enable registers and transition filters in the Device-dependent status register, the :STATus:PRESet: SCPI command is used.

If a *CLS command immediately follows a <PROGRAM MESSAGE TERMINATOR> and precedes <Query MESSAGE UNIT> elements, both the Output Queue and the MAV bit are cleared.
- Example *CLS

■ *ESE Standard Event Status Enable Command

- Parameter <DECIMAL NUMERIC PROGRAM DATA>

Integer value in the range 0 to 255

The parameter data is the sum of bit-digit values when the bits to be enabled are selected from the bits in the Standard Event Enable Register.

The register bits in this instrument are set as follows:

Bit7($2^7 = 128$)	Power on
Bit6($2^6 = 64$)	Local control execution
Bit5($2^5 = 32$)	Command error
Bit4($2^4 = 16$)	Execution error
Bit3($2^3 = 8$)	Errors which are neither command error, query error, nor execution error
Bit2($2^2 = 4$)	Query error
Bit0($2^0 = 1$)	Operation complete
- Function Sets or clears the Standard Enable Register.
- Example Set bits 2 and 4 of the Standard Enable Register.
*ESE 20

■ *ESE? Standard Event Status Enable Query

- Response <NR1 NUMERIC RESPONSE DATA>

0 to 255
- Function Queries the current value of the Standard Event Status Enable Register.
- Example A 20 is returned if *ESE? is sent after executing *ESE 20.

■ *ESR? Standard Event Status Register Query

- Response <NR1 NUMERIC RESPONSE DATA>
0 to 255
The response data is the sum of bit-digit values in the Standard Event Status Register. For setting the bits in the Standard Event Status Register, see the parameters of the *ESE command.
- Function Queries the current value of the Standard Even Status Register.
- Example A 32 is returned if a command error has occurred when queried by the :ESR? query command.

■ *SRE Service Request Enable Command

- Parameter <DECIMAL NUMERIC PROGRAM DATA>
0 to 255
The parameter data is the sum of bit-digit values when the bits to be enabled are selected from the bits in the Service Request Enable Register.
A 0 is set for the bit-digit values to be disabled.
The register bits in this instrument are set as follows:

Bit7($2^7 = 128$)	Operation status register summary
Bit6($2^5 = 32$)	Event status register summary
Bit5($2^4 = 16$)	Indicates that the output queue is not empty.
Bit4($2^3 = 8$)	Questionable status register summary
Bit3($2^2 = 4$)	Indicates that the error and event queues are not empty.
- Function Sets the bits in the service Request Enable Register.
- Example Set bit 4 of the Enable Register.
*SRE 16

■ *SRE? Service Request Enable Query

- Response <NR1 NUMERIC RESPONSE DATA>
0 to 255
- Function Queries the current value of the Service Request Enable Register.
- Example A 16 is returned if *SRE? is sent after executing *SRE 16.

■ *STB? Read Status Byte Command

- Response <DECIMAL NUMERIC PROGRAM DATA>
0 to 255
The response data is the sum of bit-digit values in the Status Byte Register. The register bits in this instrument are set as follows:

Bit7($2^7 = 128$)	Operation status register summary
Bit6($2^6 = 64$)	MSS (Master Summary Status) summary message
Bit5($2^5 = 32$)	Event status register summary
Bit4($2^4 = 16$)	Indicates that the output queue is not empty.
Bit3($2^3 = 8$)	Questionable status register summary
Bit2($2^2 = 4$)	Indicates that the error and event queues are not empty.

- **Function** Returns the current value of the status byte including the MSS (Master Summary Status) bit.
- **Example** A 32 is returned when queried by the *STB? query command if the event status register summary is TRUE.

■ *TRG **Trigger Command**

- **Parameter** None
- **Function** Has exactly the same effect as an IEEE488.1 GET (Group Execute Trigger) command.
- **Example** *TRG

■ *PSC **Power-on Status Clear Command**

- **Parameter** <DECIMAL NUMERIC PROGRAM DATA>

0	Sets the power-on-status-clear flag to FALSE.
Other than 0	Sets the power-on-status-clear flag to TRUE.
- **Function** Determines whether the Service Request Enable Register, the Standard Event Status Enable Register, and the Parallel Poll Enable Register are cleared or not at power-on.
 If a 0 is set, the enable registers are not cleared. A device can generate SRQ after power-on.
 If a 1 is set, the enable registers are cleared. A device cannot generate SRQ after power-on.
- **Example** Generate SRQ without clearing the power-on-status-clear flag.
 *PSC 0;*SRE 32;*ESE 128

■ *PSC? **Power-on Status Clear Query**

- **Response** <NR1 NUMERIC RESPONSE DATA>

0	Indicates that the power-on-status-clear flag is FALSE.
1	Indicates that the power-on-status-clear flag is TRUE.
- **Function** Queries whether the power-on-status-clear flag is TRUE or FALSE.
- **Example** *PSC?

■ *SAV **Save Command**

- **Parameter** <DECIMAL NUMERIC PROGRAM DATA>

1 to 20	
---------	--
- **Function** Stores the current state of the device in the specified location number memory. The range of the saved state is identical to the range of the state affected by the *RST command.
 The :SYStem:MEMory:STORe SCPI command has the same function as this command.
- **Example** Store the current setting state in the memory of location number 1.
 *SAV 1

■ ***RCL** **Recall Command**

- **Parameter** <DECIMAL NUMERIC PROGRAM DATA>
 0 to 20
- **Function** Recalls the state of the device from the specified memory location number to return the device to previous settings. The range of the saved state is identical to the range of the state affected by the *RST command.

The :SYSTEM:MEMory:RECall SCPI command has the same function as this command.
- **Example** Recall the data from the memory of location number 1 to make the device settings.
 *RCL 1

■ ***OPT?** **Option Identification Query**

- **Response** <ARBIRARY ASCII RESPONSE DATA>
 Character corresponding to each option

Option	Character
No options	0
Frequency offset	OPT01
MUX, DEMUX function	OPT02
Bipolar 4 ch output (Unbal)	OPT03
GPIB interface	OPT04
RS-232C interface	OPT05
Jitter function	OPT06

- **Function** Reports incorporated option lists.

The incorporated options are reported by separating them by commas.
- **Example** If the *OPT? query command is transmitted when both the MUX (Multiplexer)/DEMUX (Demultiplexer) function and the GPIB interface are incorporated, then the response data OPT02,OPT04 will be returned.

SECTION 7

STATUS REPORTING

The MP1520B status register structure conforms to the SCPI standard. (SCPI: Standard Command for Programmable Instruments; for details see Section 9.)

This section describes the status register structure and the bit definitions of the Device-dependent status register.

TABLE OF CONTENTS

7.1	MP1520B Status Register Structure	7-3
7.2	IEEE488.2-Defined Status Register	7-5
7.3	SCPI-DEfined Status Registers	7-7
7.4	MP1520B-Dependent Status Register	7-8
7.5	Reading, Writing to, and Clearing Status Registers	7-11
7.5.1	Methods of reading and writing	7-11
7.5.2	Methods of clearing and resetting	7-12

(Blank)

SECTION 7

STATUS REPORTING

7.1 MP1520B Status Register Structure

A SCPI status register structure requires that the SCPI-dependent OPERation and QUEStionable Status Registers are included in the structure defined by IEEE488.2.

The following figure shows a simplified structural diagram of the status registers incorporated in this instrument. (The bit position and length will be described later.)

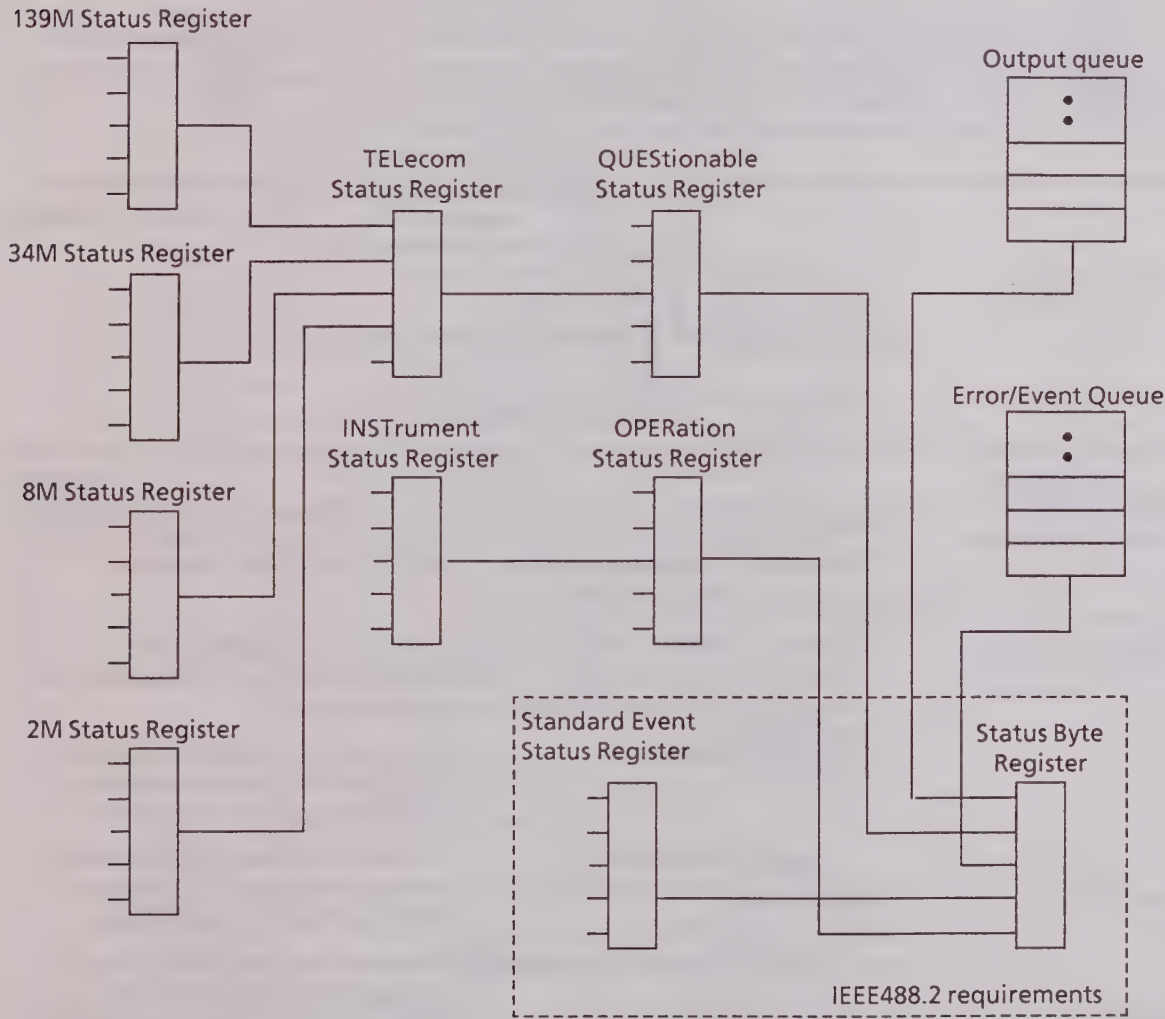


Fig. 7-1 Structural Diagram of MP1520B Status Registers

The SCPI-defined status registers and the Device-dependent status registers are configured as shown below.

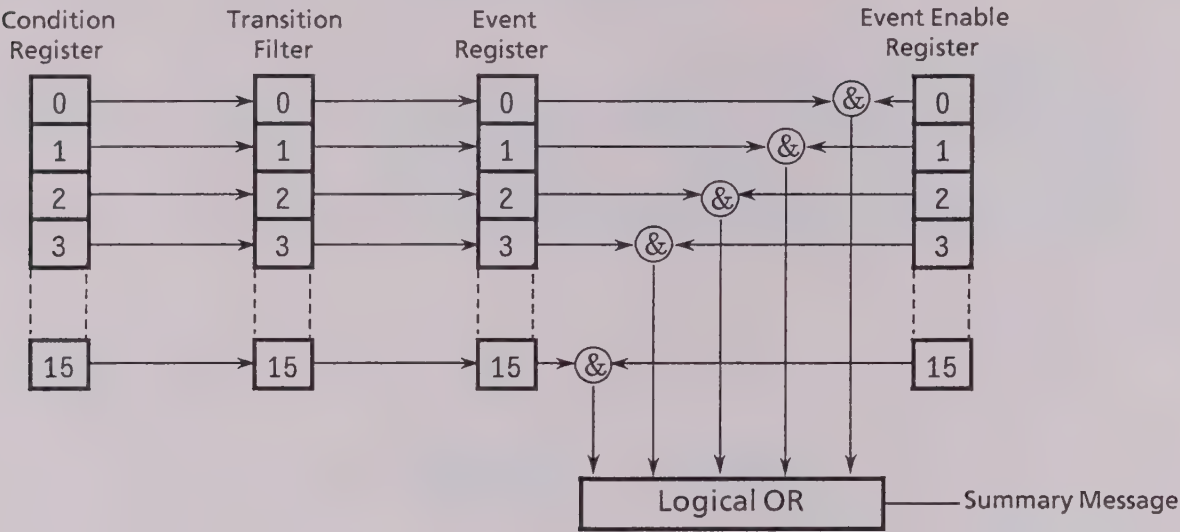


Fig. 7-2 Structural Diagram of Status Registers

Table 7-1 Contents of Register Filter

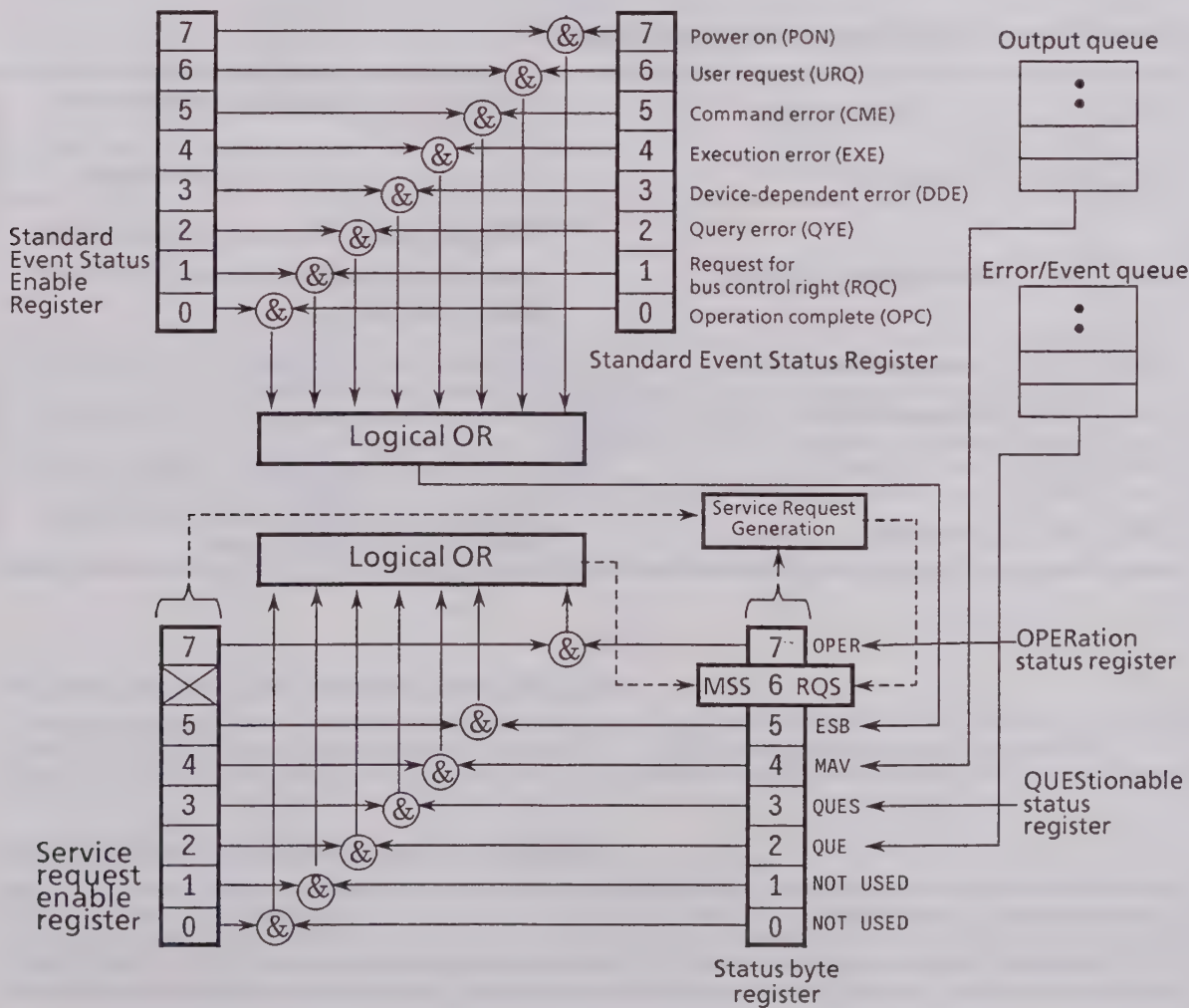
Register filter	Contents
Condition Register	The Condition Register monitors a device status, and makes real-time changes in response to that device status. This register does not memorize the status events.
Transition Filter	Sets the Condition Register contents in the Event register. The Transition Filter function selects one of the following three transitions, depending on the value in the Condition Register. Positive transition: The event becomes TRUE when its corresponding condition makes a FALSE to TRUE transition only. Negative transition: The event becomes TRUE when its corresponding condition makes a TRUE to FALSE transition only. Positive or negative transition: The event becomes TRUE when its corresponding condition makes either a FALSE to TRUE or TRUE to FALSE transition.
Event Register	Memorize the bits output from the Transition Filter.
Event Enable Register	Selects which bit in the corresponding Event Register will cause a TRUE summary message when set.

7.2 IEEE488.2-Defined Status Register

IEEE488.2 requires that the following two status registers are included.

Table 7-2 Register Contents Defined by IEEE488.2

Status Byte Register	The Status Byte Register is a register in which the RQS bit and the summary message bits can be set. It is used with the Service Request Enable Register. When the results of the OR operation of both register contents is not 0, SRQ becomes ON. Bit 6 of the Status Byte Register is reserved by the system to report that there is a service request to the external controller.
Standard Event Status Register	The Standard Event Status Register is a register in which bits are set for eight types of standard events encountered by a device. The logical OR output bit is an ESB (Event Status Bit) summary message which is represented by summarizing multiple events in a single bit on bit 5 of the Status Byte Register.



Bit Definitions of Status Byte Register

DB2	QUE (Error/Event QUEue)	Indicates that the Error/Event Queue is not empty
DB3	QUES (QUESTionable status register summary)	A summary bit from QUESTionable Status Register
DB4	MAV (Message AVailable)	Indicates that the Output Queue is not empty
DB5	ESB (Event Summary Bit)	A summary bit Standard Event Status Register
DB6	RQS (ReQuest Service) MSS (Master Summary Status)	RQS message bit: This bit indicates that there is at least one cause for a service request in the device.
DB7	OPER (OPERation status register summary)	A summary bit from OPERation Status Register

Bit definitions of Standard Event Status Register

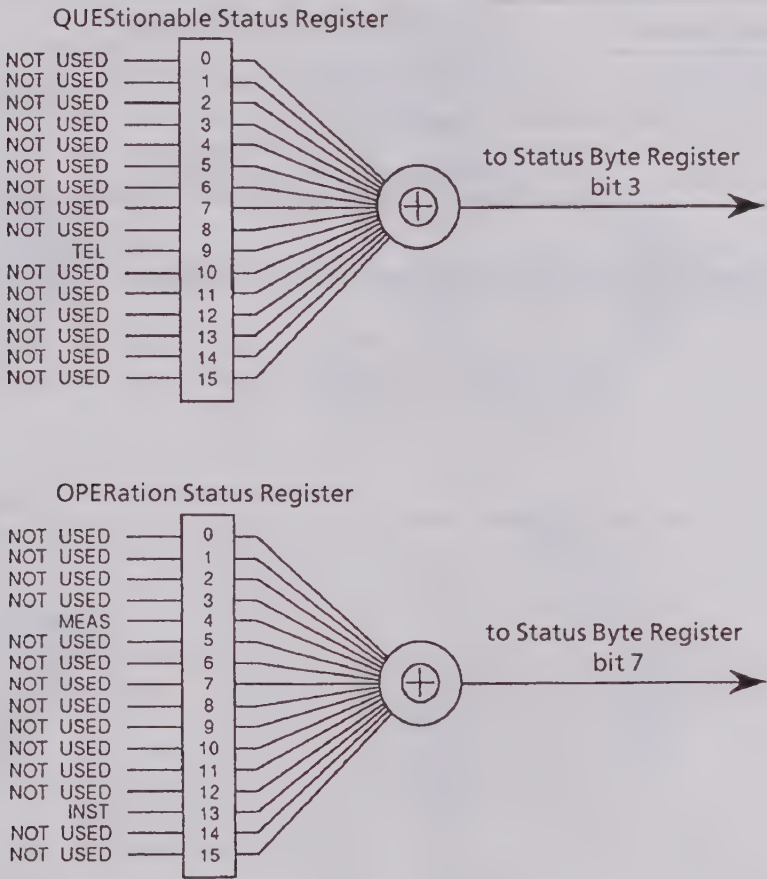
DB0	OPC (Operation Complete)	Indicates that all specified operations have been completed
DB2	QYE (Query Error)	Indicates that a query error has occurred
DB3	DDE (Device-dependent Error)	Indicates that an error has occurred which is neither a command error, a query error, nor an execution error
DB4	EXE (Execution Error)	Indicates that an execution error has occurred
DB5	CME (Command Error)	Indicates that a command error has occurred
DB6	URQ (User Request)	Indicates that local control has been executed
DB7	PON (Power on)	Indicates that an OFF-to-ON transition has occurred in the power supply

7.3 SCPI-Defined Status Registers

In addition to the registers defined by IEEE488.2, SCPI requires that the following registers are included.

Table 7-3 Register Contents Defined by SCPI

OPERation Status Register	Reports part of the instrument operation status
QUEStionable Status Register	Reports the signal status information



Bit definitions of QUEStionable Status Register

DB8	TEL (TELEcom status register summary)	TELEcom status register summary bit
-----	---------------------------------------	-------------------------------------

Bit definitions of OPERation Status Register

DB4	MEAS (MEASuring)	Shows that measurement is being executed
DB13	INST (INSTrument status register summary)	INSTrument status register summary bit

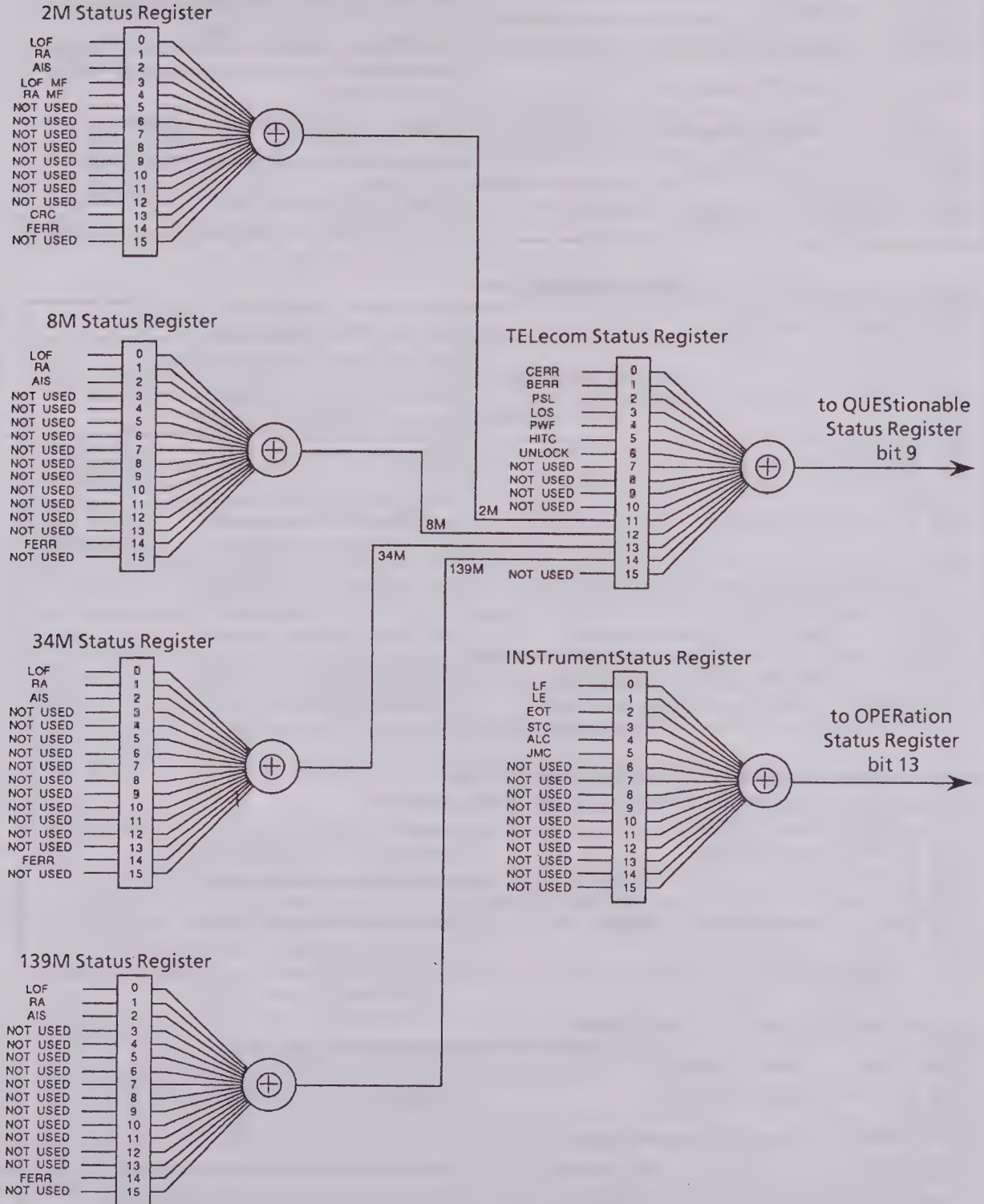
7.4 MP1520B-Dependent Status Register

The table below lists this instrument-dependent status registers.

Table 7-4 Contents of MP1520B-Dependent Register

INSTRument Status Register	Reports the operational and measurement status of this instrument
TELEcom Status Register	Reports the signal status including summary messages in a lower hierarchy register
139M Status Register	Reports the 139M signal status.
34M Status Register	Reports the 34M signal status.
8M Status Register	Reports a 8M signal status.
2M Status Register	Reports the 2M signal status.

The structural diagram of the above registers is shown on the next page.



Bit definitions of INSTRument Status Register

DB0	LF (Log Full)	Indicates that the log has become full
DB1	LE (Log Empty)	Indicates that the log is empty
DB2	EOT (End Of Test period)	Indicates that the test (measurement) has been completed
DB3	STC (Self Test Complete)	Indicates that the self-test has been completed
DB4	ALC (Alarm Change)	Indicates that an alarm transition occurred
DB5	JMC (Jitter Measurement Complete)	Indicates that the jitter measurement has been completed

Bit definitions of TELEcom Status Registers

DB0	CERR (Code ERRor)	Indicates that a code error occurred
DB1	BERR (Bit ERRor)	Indicates that a bit error occurred
DB2	PSL (Pattern Sync Loss)	Indicates that pattern sync loss has occurred
DB3	LOS (Los Of Signal)	Indicates that signal loss has occurred
DB4	PWF (PoWer Fail)	Indicates that a power failure has occurred
DB5	HITC (Hit Count)	Indicates that the Hit Count has been counted in the jitter measurement
DB6	UNLOCK (UNLOCK)	Indicates that UNLOCK has occurred
DB11	2M (2M stgatus register summary)	2M Status Register Summary
DB12	8M (8M status register summary)	8M Status Register Summary
DB13	34M (34M sttus register summary)	34M Status Register Summary
DB14	139M (139M status register summary)	139M Status Register Summary

Bit definitions of 139M/34M/8M Status Registers
 (The bit definitions are common to 139M, 34M, and 8M.)

DB0	LOF (Loss Of Frame)	Indicates that LOF has occurred
DB1	RA (Remote Alarm)	Indicates that a RA has occurred
DB2	AIS (Alarm Indication Signal)	Indicates that an AIS has occurred
DB14	FERR (Frame ERRor)	Indicates that a frame error occurred

Bit definitions of 2M Status Registers

DB0	LOF (Los Of Frame)	Indicates that LOF has occurred
DB1	RA (Remote Alarm)	Indicates that a RA has occurred
DB2	AIS (Alarm Indication Signal)	Indicates that an AIS has occurred
DB3	LOF MF (Loss Of Frame MultiFrame)	Indicates that LOF MF has occurred
DB4	RA MF (Remote Alarm MultiFrame)	Indicates that a RA MF has occurred
DB13	CRC (CRC error)	Indicates that a CRC error occurred
DB14	FERR (Frame ERRor)	Indicates that a frame error occurred

7.5 Reading, Writing to, and Clearing Status Registers

7.5.1 Methods of reading and writing

The table below shows the methods of reading the contents of the various Status Registers and writing to the registers.

Table 7-5 Methods of Reading and Writing to Status Registers

Register	Reading method	Writing method
Status Byte Register	Read by serial poll: A 7-bit status byte and RQS message bit are returned. At this time, the value of the status byte is not changed. Read by the *STB? common query: The contents of the Status Byte Register and one numeric value from the MSS (Master Summary Status) summary message are returned.	Cannot be written
Service Request Enable Register	Read by the *SRE? common query	Written to by the *SRE common command
Standard Event Status Register	Read by the *ESR? common query: At this time, the register contents are cleared after being read.	Cannot be written
Standard Event Status Enable Register	Read by the *ESE? common query: At this time, the register contents are not changed.	Written to by the *ESE common command
SCPI Event Status Register	Read by the SCPI command (STATus sub-system): :STATus:...:EVENT? At this time, the register contents are cleared.	Cannot be written
SCPI Enable Register	Read by the SCPI command (STATus sub-system): :STATus:...:ENABLE? At this time, the register contents are not changed.	Written to by the SCPI command (STATus sub-system): :STATus:...:ENABLE
SCPI Transition Filter	Read by the SCPI command (STATus sub-system): :STATus:...:PTRansition? :STATus:...:NTRansition? At this time, the register contents are not changed.	Written to by the SCPI command (STATus sub-system): :STATus:...:PTRansition :STATus:...:NTRansition
Error/Event Queue	Read by the SCPI command: :SYSTem:ERRor?	Cannot be written

Notes: • The SCPI Event Register, SCPI Enable Register, and SCPI Transition Filter correspond to the Event Register, Enable Register, and Transition Filter respectively, as used in the SCPI-defined Status Registers or Device-dependent Status Registers.

• For details of common and query commands, see Section 6.

For details of SCPI commands, see Section 9.

7.5.2 Methods of clearing and resetting

The table below shows the methods of clearing and resetting the Status Registers and the affected ranges.

Table 7-6 Methods of Clearing and Resetting Status Registers

Register	*RST	*CLS	Power-ON*	STATUS:PRESet	Other clearing methods
Status Byte Register	Unchanged	Cleared	Cleared	Unchanged	Cleared when an event is read by the *STB? query command
Service request	Unchanged	Unchanged	Cleared	Unchanged	Executes *SRE 0
Standard Event Status Register	Unchanged	Cleared	Cleared	Unchanged	Cleared when an event is read by the *ESR? query command
Standard Event Status Enable Register	Unchanged	Unchanged	Cleared	Unchanged	Executes *ESE 0
SCPI Event Register	Unchanged	Cleared	Cleared	Unchanged	Cleared when an event is read by the :STATUS:...:EVENT? Query command
SCPI Transition Filter	Unchanged	Unchanged	Reset	Reset	Executes :STATUS:...:ENABLE 0
SCPI Transition Filter	Unchanged	Unchanged	Reset	Reset	Executes :STATUS:...:PTRansition 0 or :STATUS:...:NTRansition 0
Error/Event Queue	Unchanged	Cleared	Cleared	Unchanged	All events are read by the SYSTem:ERRor? command

The table below shows the values after resetting the registers affected by the :STATUS:PRESet command.

Register	Enable/Filter	Reset value
OPERation Status Register QUESTionable Status Register	Enable Register PTRansition Filter NTRansition Filter	All 0 All 1 All 0
INSTRument Status Register	Enable Register PTRansition Filter NTRansition Filter	All 1 All 1 All 0
Other Status Registers	Enable Register PTRansition Filter NTRansition Filter	All 1 All 1 All 1

* If the PSC (Power-ON-Status-Clear) flag is TRUE with the *PSC common command, these registers are cleared or reset at power-on.

SECTION 8

GENERAL INFORMATION ON SCPI

In addition to the IEEE488.2-defined commands, the MP1520B employs SCPI (Standard Commands for Programmable Instrument) for controlling instruments remotely.

This section outlines SCPI and its command system.

In this section, and also in the section that follows, the command and response examples are indicated by > and < respectively as follows:

- > For a program message (Program command, query command)
- < For a response

TABLE OF CONTENTS

8.1	Outline	8-3
8.2	Command Structure	8-3
8.3	Writing Commands	8-4
8.4	Compounding Commands	8-5
8.5	Parameter	8-5

(Blank)

SECTION 8

GENERAL INFORMATION ON SCPI

8.1 Outline

SCPI is an instrument command language that is defined by the SCPI Consortium. It does not depend on the hardware. The goal of SCPI is to reduce the development period for Automatic Tests Equipment (ATE) programs. To accomplish this goal, SCPI provides a consistent programming environment for instrument control and data usage. It features SCPI software which is compatible between instruments of the same class, between different instruments with the same functions, and which can use the same commands and parameters to control instruments that have the same functions.

8.2 Command Structure

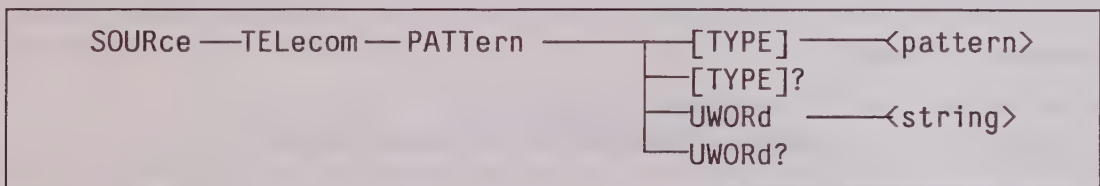


Fig. 8-1 SCPI Command Tree Example

SCPI commands are based on a hierarchical structure. The commands are grouped according to associated functions. They form hierarchical structures called 'subsystems'.

In this manual, each subsystem is represented by a command tree as shown in the above figure.

Where the same headers appear in a tree, the position of the headers correspond to different functions. This means that commands must be written with the full path including the header to be used.

Example: SOURce subsystem

SOURce denotes the highest level node.

TELEcom denotes the second level node.

PATtern denotes the third level node.

TYPE, TYPE?, UWORD, and UWORD? all denote the fourth level node.

<pattern> and <string> are used for TYPE and UWORD parameter types respectively.

8.3 Writing Commands

```

SOURCE:TELEcom:PATtern[:TYPE]<pattern>
    <pattern>=PRBS11,PRBS15,PRBS23,UWORD2,UWORD4,UWORD8,UWORD16
SOURCE:TELEcom:PATtern[:TYPE]?
SOURCE:TELEcom:PATtern:UWORD<string> <string>="00..00" to "11..11"
SOURCE:TELEcom:PATtern:UWORD?

```

Fig. 8-2 SCPI Command Example

Fig. 8-2 shows how to write the commands in the command tree shown in Fig. 8-1.

The rules for writing commands are explained below.

<Command format>

The first character of an SCPI command is the colon symbol (:).

The command is also configured by concatenating a colon (:) between headers.

<Abbreviated format for header>

A header has both a short form and a long form. A short form header is an abbreviation of a long form header. It is abbreviated to the upper-case characters of the long form header. Whether a short or long form command is used, it is interpretable as the same command with the same function. Short forms and long forms can be used together.

The command reference allows both upper-case and lower-case alphabetic characters to be used to differentiate short form keyword from the long form of the same keyword. In fact, there is no differentiation between upper-case and lower-case.

(These three types of headers, SOURCE, Source, and source are all interpretable as the same header.)

Example:	Long form>	:SOURCE:TELECOM:PATtern:TYPE PRBS11
	Short form>	:SOUR:TEL:PATT:TYPE PRBS11
	Long + short>	:SOURCE:TEL:PATtern:TYPE PRBS11

<Option node>

The symbol [] indicates an option node. The header enclosed within the [] may be abbreviated. It is accepted as a command whether abridged or unabridged.

Example:	When a header is abridged>	:SOURCE:TELECOM:PATT:TYPE PRBS11
	When a header is unabridged>	:SOURCE:TELECOM:PATT:TYPE PRBS11

<Command separator>

One or more spaces must be placed between a command and a parameter. Two or more parameters must be separated with a comma (,).

8.4 Compounding Commands

Commands can be chained with a semicolon (;) as shown in the following examples. The second command is referred to the same level as the lowest layer of the first command. This allows the second command to be written with the full path name, as shown in the example 1. However, as shown in example 2, the header with a higher order than TYPE can be omitted.

Example:

```
1>:SOURce:TELecom:PATTerN:TYPE UWORD2;;SOUR:TELecom:PATTer:UWORD "11"
2>:SOURce:TELecom:PATTerN:TYPE UWORD2;;UWORD "11"
```

8.5 Parameter

The table below shows the parameter types employed for this instrument.

In this manual these parameter types are written in lower-case alphabetical characters between brackets <>, and the IEEE488.2 (or SCPI)-defined <PROGRAM TYPE> corresponding to them is written in upper-case alphabetical characters. The correspondence between each parameter and IEEE488.2 (or SCPI) is written for the various commands.

Table 8-1 Contents of Parameter Types

Parameter type	Description
<numeric> <DECIMAL NUMERIC PROGRAM DATA>	Represents decimal numbers. A <CHARACTER PROGRAM DATA> item, such as MINimum or MAXimum, is included in a special numeric format. An integer format is employed for almost all numeric values used by this instrument. The fraction part is rounded off.
<boolean> <BOOLEAN PROGRAM DATA> (defined by SCPI)	Represents a theoretical value. OFF or 0 corresponds to FALSE, and ON or 1 corresponds to TRUE. When 0, 1 or OFF, ON are used for setting, queries return 1 or 0, never ON or OFF.
<string> <STRING PROGRAM DATA>	Represents a character string. Both ends of a character string have single quotation marks or double quotation marks. Both long and short forms can be used. Example: 'LOF:M139' or "LOF:M139"
<brate>, <type>..etc. <CHARACTER PROGRAM DATA>	Represents a character data. It is possible to express a short character string corresponding to the setting contents. Both long and short forms can be used.

(Blank)

SECTION 9

DEVICE MESSAGES

This section describes in detail the device messages which are supported by the MP1520B.

In this section, the command and response examples are indicated by > and < respectively as follows:

> For a program message (Program command, query command)

< For a response

TABLE OF CONTENTS

9.1	Response Format	9-3
9.2	Buffer Size Requirements	9-6
9.3	Device-dependent Commands	9-7
9.3.1	INSTRument subsystem	9-8
9.3.2	SOURce subsystem	9-9
9.3.3	SENSe subsystem	9-26
9.3.4	DISPlay subsystem	9-43
9.3.5	ROUTe subsystem	9-59
9.3.6	CALCulate subsystem	9-60
9.3.7	SYSTem subsystem	9-70
9.3.8	TEST subsystem	9-82
9.3.9	STATus subsystem	9-84

(Blank)

SECTION 9 DEVICE MESSAGES

9.1 Response Format

This paragraph describes the format of the response message returned in response to a query command. The response messages are returned according to the formats shown in the following tables.

Table 9-1 Response Format (by response type)

Response	Format
<numeric>,<year>,... (<NR1 NUMERIC RESPONSE DATA>)	The number of digits responded varies up to a maximum equal to the maximum number of digits in the numeric value range. There is no space between a sign and a number. > :SYSTem:DATA? < 1993, 7, 14
<brate>,<type> ...etc (<CHARACTER RESPONSE DATA>)	A short form character is returned. Example: > :SYSTem:BEEPer:TYPE? < ERR
<string>,<display> ...etc (<STRING RESPONSE DATA>)	A character string enclosed with double quotation marks (") is returned. If there are both short form and long form values in the contents of a corresponding program command with a string, the short form shall be returned. > :DISPlay:MENU:NAME? < "ANAL:TGR" (short form of "ANALysis: TGRaph") For Form 1 to Form 11 response formats, see the following table.

Table 9-2 Response Format (Form 1 to Form 11) (1/4)

Form	Format	Description
Form 1 Integer type	" XXXXXX"	If $0 \leq \text{Value} \leq 999,999$, six characters are aligned to the right within an nine character space. Example: > :CALCulate:DATA? "EC:BIT" < " 892"
	" X.XXE+XX"	If $1.00\text{E} + 06 \leq \text{Value} \leq 9.99\text{E} + 15$, eight characters are aligned to the right within an nine character space. Example: > :CALCulate:DATA? "EC:BIT" < " 1.25E+07"
	">9.99E+15"	Where $1.00\text{E} + 16 \leq \text{Value}$
	"-----"	Where there is no data corresponding to the query Example: The measured value for 139 Mb/s has been queried when an input signal is 8 Mb/s.

Table 9-2 Response Format (Form 1 to Form 11) (2/4)

Form	Format	Description
Form 2 Fractional type	" 0.00E-XX"	If value = 0, eight characters are aligned to the right within an nine character space.
	"<1.00E-15"	Where value $\leq 9.99E - 16$
	" X.XXE-XX"	If $1.00E - 15 \leq \text{value} \leq 9.99E - 01$, eight characters are aligned to the right within an nine character space. Example: > :CALCulate:DATA? "ER:BIT" < " 1.00E-05"
	" 1.00E-00"	If value = 1, eight characters are aligned to the right within an nine character space.
	"-----"	Where there is no data corresponding to the query
Form 3 % type	"XXX.XXXX"	0.0000~100.0000 Example: > :CALCulate:DATA? "G821:ES:BIT" < " 1.0010"
	"-----"	Where here is no data corresponding to the query
Form 4 ppm type	">+100.0"	Where value > 100
	" +XXX.X"	If value > 0, six characters are aligned to the right within a seven character space, Example: > :CALCulate:DATA? "FREQuency:OFFSet" < " + 12.0"
	" 0.0"	If value = 0, three characters are aligned to the right within a seven character space.
	" -XXX.X"	If value < 0, seven characters are aligned to the right within a seven character space. Example: > :CALCulate:DATA? "FREQuency:OFFSet" < " - 12.0"
	"<-100.0"	Where value < -100
	"-----"	Where there is no data corresponding to the query
Form 5 Decision type	" Acceptable"	Represents a character string so that it is aligned to the right within a 12 character space.
	" Degraded"	
	" Unacceptable"	
	"-----"	Where here is no data corresponding to the query

Table 9-2 Response Format (Form 1 to Form 11) (3/4)

Form	Format	Description
Form 6	" 1 Line Data"	Transverse line of 38 characters is taken as one character string when log data is displayed on the screen. Example: Letting :CALCulate:LOG:DATA? 3,5 (a) Where there is all data "Measuring Condition (Out-of-Service)", " Bit rate: 139Mb/s", " Patt : Word16[0101110000111010]" (NL^END) (b) Where there is no data on the fifth line "Measuring Condition (Out-of-Service)", " Bit rate: 139Mb/s", (NL^END) (c) Where there is no data (NL^END)
Form 7 UI type	" X.XXX" (1) " XX.XX" (2) " XX.XXX"(3)	$0.000 \leq \text{Value} \leq 2.020$ or 20.20 • The format (1) applies for 2UI range, (2) for 20UI range, and (3) for the Jitt/Freq measurement result. Example: > :CALCulate:DATA? "RX:AMPLitude" > " 1.234"
	">X.XXX" (1) ">XX.XX" (2) ">XX.XXX"(3)	When overflow occurs • The format (1) applies for 2UI range, (2) for 20UI range, and (3) for the Jitt/Freq measurement result. The value X differs depending on the measurement result queried.
	"-----" (1) "-----"(2)	Where there is no data corresponding to the query • (2) for the result of Jitt/Freq measurement and (1) for others.
Form 8 dB type	" X.XX"	Value > 0 Example: > :CALCulate:DATA? "CTABle:POINT1" < " 1.23"
	" 0.00"	Value = 0
	" -XX.XX"	Value < 0
	"> 9.99" "< -29.99" " **.*"	Overflow and Underflow
	"-----"	When there is no data corresponding to the query Example: When the measurement result of point 2 is queried upon the completion of Transfer measurement of point 1.

Table 9-2 Response Format (Form 1 to Form 11) (4/4)

Form	Format	Description
Form 9 ppm type 2	" +XX"	Value > 0 Example: > :CALCulate:DATA? "MAX:OFFSet" < "+36"
	" 0"	Value = 0
	" -XX"	Value < 0
	" ---"	When there is no data corresponding to the query
Form 10 Decision type 1	"UNLOCK"	When UNLOCK occurs
	(NL^END)	When UNLOCK does not occur
	"-----"	When there is no data corresponding to the query Example: When the Status of Sweep mask measurement is queried when no measurement is being performed.
Form 11 Decision type 2	" Error"	When an error occurs in the measurement of the point corresponding to the query in Sweep mask measurement.
	"No Error"	When an error does not occur in the measurement of the point corresponding to the query in Sweep mask measurement.
	"-----"	When there is no data corresponding to the query

9.2 Buffer Size Requirements

The response messages described above are sent from the device to the controller. To accomplish data transfer successfully, the controller's buffer memory size must be large enough to accept response messages correctly.

Table 9-3 Requirements for Buffer Size

For inputting log data into buffer :CALCulate:LOG:DATA?	41 bytes x Specified linkage 41 bytes of buffer memory are required for inputting one line of log data. The number of log data lines that can be read is restricted by the buffer size that can be reserved in the controller. Example: Letting :CALCulate:LOG:DATA? 3,5 the necessary buffer size is as follows: $41 \times 3 = 123$ bytes
Other commands	60 bytes If a maximum of 60 bytes can be reserved, the buffer size is large enough to use other commands.

9.3 Device-dependent Commands

From this paragraph on, the device-dependent commands are described in detail. Since the device-dependent commands are written according to subsystem classification, for the correspondence with the screen, see Appendix D Correspondence Between Commands and Screen. All of the device-dependent commands supported by this instrument are sequential commands.

If a program command setting value does not allow other items to be set, the program command setting value is changed to the allowed value.

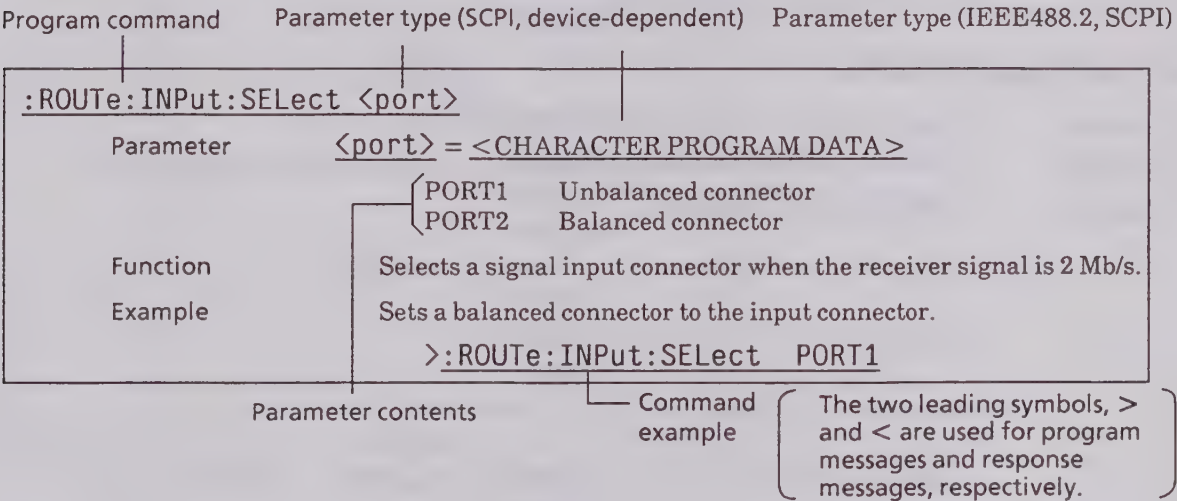
If the program setting value is changed during measurement, then the measurement is restarted.

To change the setting values of other items and to restart the measurement, refer to the Detailed Operating Instructions in the separate MP1520B Operation Manual.

The command description examples are shown below.

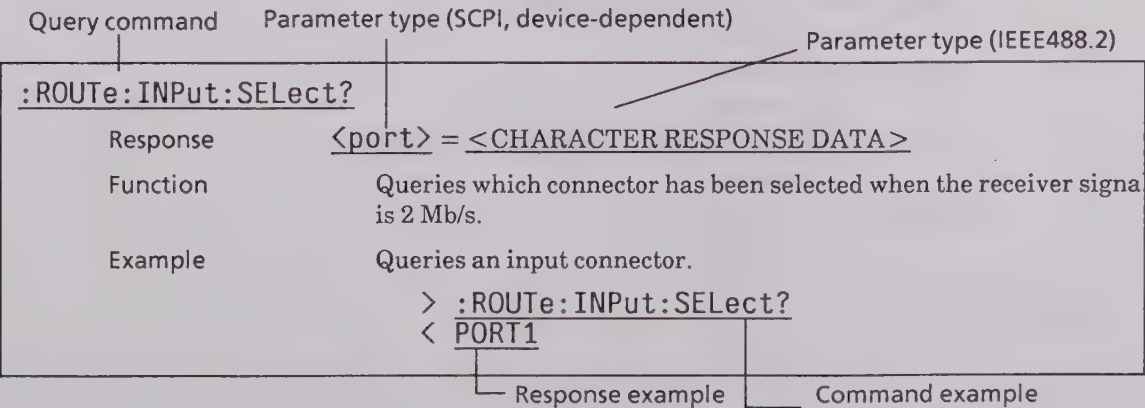
■ <Program command>

If there are restrictions when using the command, in addition to the example below, the item 'Limitation' is provided.



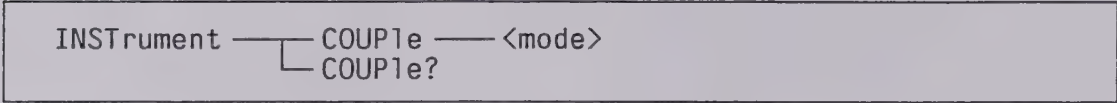
■ <Program query command>

The response contents are abbreviated if there is a program command corresponding to the query. (Same as the parameter contents of program command)



9.3.1 INSTRument subsystem

The INSTRument subsystem is used to make transmitter and receiver settings either jointly or individually.



■ :INSTRument:COUPle <mode>

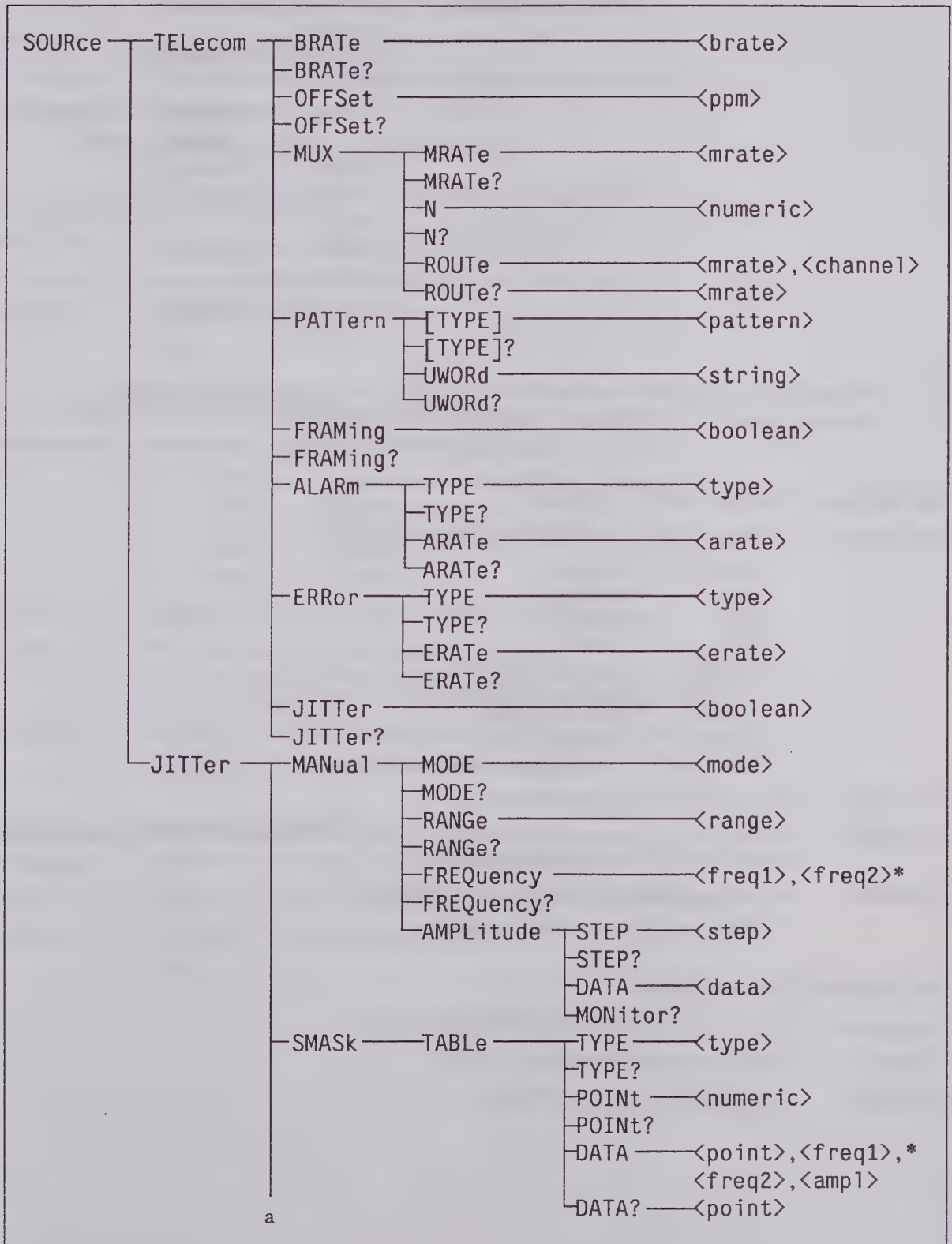
- Parameter <mode> = <CHARACTER PROGRAM DATA>
 ALL Tx&Rx
 NONE Individual
- Function Switches a transmitter and a receiver to make either joint, or individual settings.
- Example Set the transmitter and receiver settings jointly.
 > :INSTRument:COUPle ALL

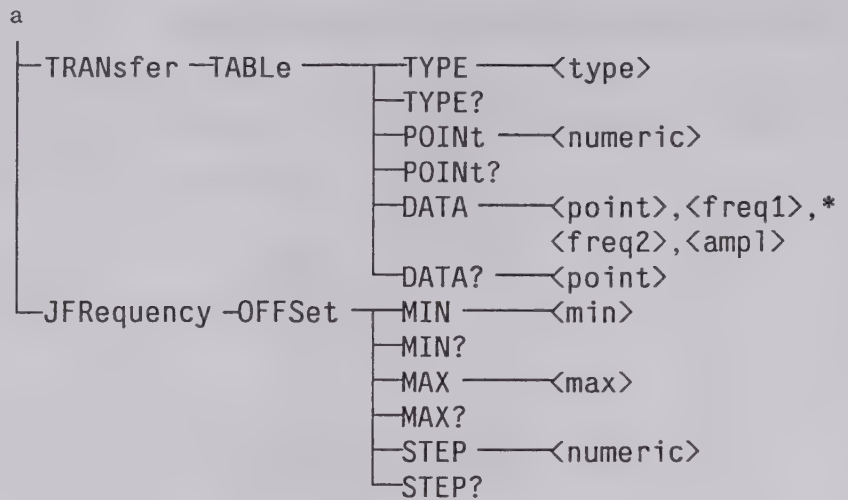
■ :INSTRument:COUPle?

- Response <mode> = <CHARACTER RESPONSE DATA>
- Function Queries whether transmitter and receiver settings have been set jointly or not.
- Example > :INSTRument:COUPle?
 < ALL

9.3.2 SOURce subsystem

The SOURce subsystem is used to make transmitter settings.





* It is possible to set a number expressed in real number format using these command.
 (See paragraph 4.3.2 <DECIMAL NUMERIC PROGRAM DATA>)

■ :SOURce:TELEcom:BRATe <brate>

- Parameter <brate>= <CHARACTER PROGRAM DATA>

M139	139 Mb/s
M34	34 Mb/s
M8	8 Mb/s
M2	2 Mb/s
EXTM139	External 139 Mb/s
EXTM34	External 34 Mb/s
EXTM8	External 8 Mb/s
EXTM2	External 2 Mb/s
RECeive	Synchronizes with a receiver signal.

- Function Sets the transmitter signal bit rate.
- Limitation If the <ALL> parameter is selected for the :INSTrument:COUPle command, it makes the <RECeive> setting invalid.
- Example Set the transmitter signal bit rate to 139 Mb/s.
 > :SOURce:TELEcom:BRATe M139

■ :SOURce:TELEcom:BRATe?

- Response <brate>= <CHARACTER RESPONSE DATA>
- Function Queries the transmitter signal bit rate.
- Example > :SOURce:TELEcom:BRATe?
 <M139

■ :SOURce:TELecom:OFFSet <ppm>

- Parameter <ppm>= <DECIMAL NUMERIC PROGRAM DATA>
 -100 to 100 Step value: 1
- Function Sets the transmitter signal offset value.
- Limitation Not valid unless the frequency offset option is incorporated.
- Example Set the frequency offset value to 100 ppm.
 > :SOURce:TELecom:OFFSet 100

■ :SOURce:TELecom:OFFSet?

- Response <ppm>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the frequency offset value of the transmitter
- Example > :SOURce:TELecom:OFFSet?
 < 100

■ :SOURce:TELecom:MUX:MRATe <mrte>

- Parameter <mrte>= <CHARACTER PROGRAM DATA>

OFF	Does not use the Multiplexer function
M34	34 Mb/s
M8	8 Mb/s
M2	2 Mb/s
K64	6 4kb/s
- Function Sets the final tributary signal bit rate in the Multiplexer.
- Limitation Not valid in the following cases:
 - The Multiplexer Option is not incorporated
 - A higher order bit rate than interface speed is set
- Example Set the final tributary bit rate in the Multiplexer to 2 Mb/s.
 > :SOURce:TELecom:MUX:MRATe M2

■ :SOURce:TELecom:MUX:MRATe?

- Response <mrte>= <CHARACTER RESPONSE DATA>
- Function Queries the bit rate of the final tributary signal in the Multiplexer.
- Example > :SOURce:TELecom:MUX:MRATe?
 < M2

■ :SOURce:TELEcom:MUX:N <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 31 Step value: 1
- Function Sets consecutive channels of 64 kb/s.
 Inserts a test signal whose N channels are consecutive from the 64 kb/s channel specified in the :SOURce:TELEcom:MUX:ROUTE command.
- Limitation Not valid unless the final tributary bit rate is specified up to 64 kb/s.
 Up to <31> can be set only when :SENSe:TELEcom:M2:MChannel is <31>.
- Example Set the consecutive channels to 5.
 > :SOURce:TELEcom:MUX:N 5

■ :SOURce:TELEcom:MUX:N?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the set value of consecutive channel at 64 kb/s.
- Example > :SOURce:TELEcom:MUX:N?
 < 5

■ :SOURce:TELEcom:MUX:ROUTE <mrate>,<channel>

- Parameter <mrate>= <CHARACTER PROGRAM DATA>

M34	34 Mb/s
M8	8 Mb/s
M2	2 Mb/s
K64	64 kb/s

 <channel>= <DECIMAL NUMERIC PROGRAM DATA>

1 to 4	When <mrate> is set to M34, M8, or M2
1 to 31-N	When <mrate> is set to K64, Step value: 1 (Only 1 can be set for N=31.)

N: The value specified in :SOURce:TELEcom:MUX:N command
- Function Sets the channels for each stage in the Multiplexer.
- Limitation Not valid in the following cases:
 - ▶ The Multiplexer Option is not incorporated.
 - ▶ The <OFF> parameter is set in the :SOURce:TELEcom:MUX:MRATE command.
 - ▶ A higher order bit rate than interface speed is set.
 - ▶ A lower order speed than the bit rate set in the :SOURce:TELEcom:MUX:MRATE command is set.
- Example Set the channel number at 8 Mb/s to 4.
 > :SOURce:TELEcom:MUX:ROUTE M8, 4

■ :SOURce:TELEcom:MUX:ROUTe? <mrate>

- Parameter <mrate>= <CHARACTER PROGRAM DATA>
- Response <channel>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the channel which has been set in the Multiplexer.
- Example Query the channel which has been set at 8 Mb/s.
 > :SOURce:TELEcom:MUX:ROUTe? M8
 < 4

■ :SOURce:TELEcom:PATtern[:TYPE] <pattern>

- Parameter <pattern>= <CHARACTER PROGRAM DATA>
 - PRBS11 PRBS2¹¹-1
 - PRBS15 PRBS2¹⁵-1
 - PRBS23 PRBS2²³-1
 - UWORD2 The word data is 2 bits in length.
 - UWORD4 The word data is 4 bits in length.
 - UWORD8 The word data is 8 bits in length.
 - UWORD16 The word data is 16 bits in length.
- Function Sets the test pattern for the transmitter signal.
- Example Set the test pattern to the word data that is 16 bits in length.
 > :SOURce:TELEcom:PATtern:TYPE UWORD16
 or :SOURce:TELEcom:PATtern UWORD16 (TYPE can be abbreviated)

■ :SOURce:TELEcom:PATtern[:TYPE]?

- Response <pattern>= <CHARACTER RESPONSE DATA>
- Function Queries the test pattern which has been set for the transmitter signal.
- Example > :SOURce:TELEcom:PATtern:TYPE?
 or :SOURce:TELEcom:PATtern? (TYPE can be abbreviated)
 < UWORD16

■ :SOURce:TELEcom:PATtern:UWORD <string>

- Parameter <string>= <STRING PROGRAM DATA>
 "00...0" to "11...1"
- Function Sets the test pattern word data for the transmitter signal.
- Limitation ▶ The number of digits in a string corresponds to the word data length set in the
 :SOURCE:TELEcom:PATtern[:TYPE] command.
 ▶ Not valid if the <PRBS11, 15, 23> parameter is selected for the
 :SOURCE:TELEcom:PATtern[:TYPE] command.
- Example Set the word data of the 8-bit word length to "11001100".
 > :SOURce:TELEcom:PATtern:UWORD "11001100"

■ :SOURce:TELecom:PATtern:UWORD?

- Response <string>= <STRING RESPONSE DATA>
- Function Queries the test pattern word data which has been set for the transmitter signal.
- Example > :SOURce:TELecom:PATtern:UWORD?
< "11001100"

■ :SOURce:TELecom:FRAMing <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>

OFF or 0	Without frame
ON or 1	With frame
- Function Switches the test signal frame to ON or OFF for the transmitter signal.
Switches the frame of the final tributary signal to ON or OFF when the Multiplexer (Option 02) is turned ON.
- Example Set the test signal frame to ON.
> :SOURce:TELecom:FRAMing ON

■ :SOURce:TELecom:FRAMing?

- Response <NR1 NUMERIC RESPONSE DATA>
0 or 1
- Function Queries the test pattern frame which has been set for the transmitter signal.
Queries the frame of the final tributary signal when the Multiplexer (Option 02) is turned ON.
- Example > :SOURce:TELecom:FRAMing?
< 1

■ :SOURce:TELecom:ALARm:TYPE <type>

- Parameter <type>= <CHARACTER PROGRAM DATA>

OFF	Does not insert alarm.
AIS139	Inserts AIS in a 139 Mb/s signal.
AIS34	Inserts AIS in a 34 Mb/s signal.
AIS8	Inserts AIS in a 8 Mb/s signal.
AIS2	Inserts AIS in a 2-Mb/s signal.
FAS139	Inserts the FAS alarm in a 139 Mb/s signal.
FAS34	Inserts the FAS alarm in a 34 Mb/s signal.
FAS8	Inserts the FAS alarm in a 8 Mb/s signal.
FAS2	Inserts the FAS alarm in a 2 Mb/s signal.
RA139	Inserts RA in a 139 Mb/s signal.
RA34	Inserts RA in a 34 Mb/s signal.
RA8	Inserts RA in a 8 Mb/s signal.
RA2	Inserts RA in a 2 Mb/s signal.
RAMF	Inserts RAMF in a 2 Mb/s signal.
- Function Sets any one of the alarm types to be inserted in the specified transmitter signal.

- **Limitation** Not valid in the following cases:
 - ▶ The alarm insertion is set to a signal with a lower order bit rate than the final tributary in the Multiplexer
 - ▶ The alarm insertion is set to a signal with a higher order bit rate than the interface
 - ▶ The final tributary FAS alarm or remote alarm is set with the <OFF> parameter set in the :SOURce:TELEcom:FRAMing command
 - ▶ The <RAMF> parameter is set with the <OFF> parameter selected for the :SENSe:TELEcom:M2:SIGNAling command.
- **Example** Insert AIS in the 8 Mb/s signal.
 >:SOURce:TELEcom:ALARm:TYPE AIS8

■ :SOURce:TELEcom:ALARm:TYPE?

- **Response** <type>= <CHARACTER RESPONSE DATA>
- **Function** Queries the alarm type which has been selected to be inserted in the specified transmitter signal.
- **Example** >:SOURce:TELEcom:ALARm:TYPE?
 < AIS8

■ :SOURce:TELEcom:ALARm:ARATe <arate>

- **Parameter** <arate>= <CHARACTER PROGRAM DATA>

A1_16	1/16
A2_16	2/16
A3_16	3/16
A4_16	4/16
- **Function** Sets the rate for inserting FAS alarm in the transmitter signal.
- **Limitation** Not valid if the parameter other than <FAS> is set in the SOURce:TELEcom:ALARm:TYPE command.
- **Example** Insert the FAS alarm at a rate of 1/16.
 >:SOURce:TELEcom:ALARm:ARATe A1_16

■ :SOURce:TELEcom:ALARm:ARATe?

- **Response** <arate>= <CHARACTER RESPONSE DATA>
- **Function** Queries the rate of the FAS alarm inserted in the transmitter signal.
- **Example** >:SOURce:TELEcom:ALARm:ARATe?
 < A1_16

■ :SOURce:TELEcom:ERRor:TYPE <type>

- Parameter <type>= <CHARACTER PROGRAM DATA>

OFF	Does not insert an error.
BIT139	Inserts a bit error in a 139 Mb/s signal.
BIT34	Inserts a bit error in a 34 Mb/s signal.
BIT8	Inserts a bit error in a 8 Mb/s signal.
BIT2	Inserts a bit error in a 2 Mb/s signal.
CODE	Inserts a code error.

- Function Sets any one of the error types to be inserted in the specified transmitter signal.

- Limitation Not valid in the following cases:

- ▶ The error insertion is set to signal with lower order bit rate than the final tributary in the Multiplexer
- ▶ The error insertion is set to a signal with a higher order bit rate than the interface

- Example Insert a bit error in a 34 Mb/s signal.

```
> :SOURce:TELEcom:ERRor:TYPE    BIT34
```

■ :SOURce:TELEcom:ERRor:TYPE?

- Response <type>= <CHARACTER RESPONSE DATA>

- Function Queries the error type which has been selected to be inserted in the specified transmitter signal.

- Example > :SOURce:TELEcom:ERRor:TYPE?

```
< BIT34
```

■ :SOURce:TELEcom:ERRor:ERATe <erate>

- Parameter <erate>= <CHARACTER PROGRAM DATA>

E_3	1E-3
E_4	1E-4
E_5	1E-5
E_6	1E-6
E_7	1E-7
ONCE	Single error

- Function Sets the rate for inserting errors in the transmitter signal.

- Limitation Not valid if the <OFF> parameter is set in the :SOURce:TELEcom:ERRor:TYPE command.

- Example Insert errors at the rate of 1E-3

```
> :SOURce:TELEcom:ERRor:ERATe    E_3
```

■ :SOURce:TELEcom:ERRor:ERATe?

- Response <erate>= <CHARACTER RESPONSE DATA>

- Function Queries the rate for inserting errors in the transmitter signal.

- Example > :SOURce:TELEcom:ERRor:ERATe?

```
< E_3
```

■ :SOURce:TELeom:JITter <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Does not generate a jitter.
 ON or 1 Generates a jitter.
- Function Sets whether the jitter in the transmitter is being generated or not.
- Limitation Not valid unless the jitter option is incorporated.
- Example Generates a jitter.
 > :SOURce:TELeom:JITter ON

■ :SOURce:TELeom:JITter?

- Response <NR1 NUMERIC RESPONSE DATA>
- Function Queries whether a jitter is being generated in the transmitter.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SOURce:TELeom:JITter?
 < 1

■ :SOURce:JITter:MANual:MODE <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>
 Internal Modulates the jitter by internal signal.
 EHIGH Modulates the jitter by eternal signal of 10 Hz or more.
 ELow Modulates the jitter by external signal of less than 10 Hz.
- Function Switches jitter modulation by internal signal or by external signal.
- Limitation Not valid unless the jitter option is incorporated.
- Example Modulates the jitter by internal signal.
 > :SOURce:JITter:MANual:MODE Internal

■ :SOURce:JITter:MANual:MODE?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries whether jitter is being modulated by internal signal or external signal.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SOURce:JITter:MANual:MODE?
 < INT

■ :SOURce:JITTer:MANual:RANGe <range>

- Parameter <range>= <CHARACTER PROGRAM DATA>

UI20	Sets the range to 20UI.
UI2	Sets the range to 2UI.
- Function Sets the jitter amplitude range.
- Limitation Not valid unless the jitter option is incorporated.
- Example Sets the jitter amplitude to 20UI.
 >:SOURce:JITTer:MANual:RANGe UI20

■ :SOURce:JITTer:MANual:RANGe?

- Response <range>= <CHARACTER RESPONSE DATA>
- Function Queries the jitter amplitude range.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:SOURce:JITTer:MANual:RANGe?
 < UI20

■ :SOURce:JITTer:MANual:FREQuency <freq1>,<freq2>

- Parameter <freq1>= <DECIMAL NUMERIC PROGRAM DATA>

1.0 to 990.0	Step value: 0.1
--------------	-----------------

 <freq2>= <CHARACTER PROGRAM DATA>

HZ	Sets the unit of <freq1> to Hz.
KHZ	Sets the unit of <freq1> to kHz.
MHZ	Sets the unit of <freq1> to MHz.
- Function Sets the jitter modulation frequency.
- Limitation The parameter may be set to 1.0 Hz to 990.0 Mhz regardless of other settings. If the specified value is outside the setting range, the value is changed to one that is within the range and set. (Refer to the Panel operating instructions manual for the setting range.)
 Not valid in the following cases:
 - ▶ When the jitter option is not incorporated.
 - ▶ When :SOURce:JITTer:MANual:MODE is not set to <INTernal>.
- Example The jitter modulation frequency is set to 1.3 kHz.
 >:SOURce:JITTer:MANual:FREQuency 1.3,KHZ

■ :SOURce:JITTer:MANual:FREQuency?

- Response <freq1>,<freq2>
 <freq1>= <NR2 NUMERIC RESPONSE DATA>
 <freq2>= <CHARACTER RESPONSE DATA>
- Function Queries the jitter modulation frequency.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:SOURce:JITTer:MANual:FREQuency?
 < 1.3,KHZ

■ :SOURce:JITTer:MANual:AMPLitude:STEP <step>

- Parameter <step>= <CHARACTER PROGRAM DATA>
 FINE Sets the step to Fine.
 COARSe Sets the step to Coarse.
- Function Sets the step for setting the jitter amplitude.
- Limitation Not valid in the following cases:
 - When the jitter option is not incorporated.
 - When :SOURce:JITTer:MANual:MODE is not set to <INTernal>.
- Example Sets the step to Fine.
 >:SOURce:JITTer:MANual:AMPLitude:STEP FINE

■ :SOURce:JITTer:MANual:AMPLitude:STEP?

- Response <step>= <CHARACTER RESPONSE DATA>
- Function Queries the step for setting the jitter amplitude.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:SOURce:JITTer:MANual:AMPLitude:STEP?
 < FINE

■ :SOURce:JITTer:MANual:AMPLitude:DATA <data>

- Parameter <data>= <CHARACTER PROGRAM DATA>
 UP Increases the jitter amplitude.
 DOWN Decreases the jitter amplitude.
- Function Sets the jitter amplitude.
- Limitation Not valid in the following cases:
 - When the jitter option is not incorporated.
 - When :SOURce:JITTer:MANual:MODE is not set to <INTernal>.
- Example Increases the jitter amplitude.
 >:SOURce:JITTer:MANual:AMPLitude:DATA UP

```
■ :SOURCE:JITTer:MANUal:AMPLitude:MONitor?
```

- | | |
|--------------|---|
| ● Response | <p><STRING RESPONSE DATA></p> <p>" XX.XX" (Tx range:20UI)</p> <p>" X.XXX" (Tx range: 2UI)</p> <p>Refer to response format Form 7.</p> |
| ● Function | Queries the actually set jitter amplitude. |
| ● Limitation | Not valid unless the jitter option is incorporated. |
| ● Example | <pre>> :SOURce:JITTer:MANual:AMPLitude:MONitor? < " 10.00"</pre> |

■ :SOURce:JITTer:SMASk:TABLE:TYPE <type>

- | | | |
|--------------|---|---|
| ● Parameter | <type> = <CHARACTER PROGRAM DATA> | |
| | R | Sets to the fixed table conforming to ITU-T G.823. |
| | N | Sets to the fixed table conforming to ITU-T G.823 National network. |
| | U | Sets to a table that can be set arbitrarily. |
| ● Function | Sets the type of table used for the Sweep mask measurement. | |
| ● Limitation | Not valid unless the jitter option is incorporated. | |
| | <N> can be set only when :SENSe:TELEcom:BRATe is <M8> and <M2>. | |
| | This command sets the type of bit rate table concurrently selected. When setting a different type of bit rate table, change the bit rate before executing this command. | |
| ● Example | Sets the bit rate 2 Mb/s table to N. | |
| | > :SENSe:TELEcom:BRATe M2 | ←Sets the bit rate to 2 Mb/s. |
| | > :SOURce:JITTer:SMASK:TABLE:TYPE N | ←Sets the 2M table to N. |

```
■ :SOURce:JITTer:SMASk:TABLE:TYPE?
```

- | | | |
|--------------|---|-------------------------------|
| ● Response | <type>=<CHARACTER RESPONSE DATA> | |
| ● Function | Queries the type of table used for the Sweep mask measurement. | |
| ● Limitation | Not valid unless the jitter option is incorporated. | |
| | This query queries the type of bit rate table concurrently selected. When querying a different type of bit rate table, change the bit rate before executing this command. | |
| ● Example | Queries the type of bit rate 2 Mb/s table. | |
| | > :SENSe:TELEcom:BRATe M2 | ←Sets the bit rate to 2 Mb/s. |
| | > :SOURce:JITTer:SMASk:TABLE:TYPE? | ←Queries the type of table. |
| | < N | ←2M table type is returned. |

■ :SOURce:JITTer:SMASk:TABLE:POINt <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 10 Step value: 1
- Function Sets the number of Sweep mask measuring points.
- Limitation Not valid unless the jitter option is incorporated.
 Valid only when :SOURce:JITTer:SMASk:TABLE:TYPE is <U>.
- Example Sets the number of measuring points to 5.
 > :SOURce:JITTer:SMASk:TABLE:POINt 5

■ :SOURce:JITTer:SMASk:TABLE:POINt?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the number of Sweep mask measuring points.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SOURce:JITTer:SMASk:TABLE:POINt?
 < 5

■ :SOURce:JITTer:SMASk:TABLE:DATA <point>,<freq1>,<freq2>,<amp1>

- Parameter <point>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 10 Step value: 1
 <freq1>= <DECIMAL NUMERIC PROGRAM DATA>
 1.0 to 990.0 Step value: 0.1
 <freq2>= <CHARACTER PROGRAM DATA>
 HZ Sets the unit of <freq1> to Hz.
 KHZ Sets the unit of <freq1> to kHz.
 MHZ Sets the unit of <freq1> to MHz.
 <amp1>= <DECIMAL NUMERIC PROGRAM DATA>
 0.000 to 20.000 Step value: 0.001
- Function Sets the jitter modulation frequency and jitter amplitude against the output signal upon Sweep mask measurement.
- Limitation Not valid in the following cases:
 - ▶ When the jitter option is not incorporated.
 - ▶ When :SOURce:JITTer:SMASk:TABLE:TYPE is not set to <U>.
 - ▶ When the <point> value is larger than the set value of :SOURce:JITTer:SMASk:TABLE:POINt.

<freq1>, <freq2>, and <amp1> may be specified regardless of other settings. If the specified value is outside the setting range, the value is changed to one that is within the range and set. (Refer to the Panel operating instructions manual for the setting range.)
- Example Sets point 1 to 12.0 Hz 13.000 UI_{p.p.}
 > :SOURce:JITTer:SMASk:TABLE:DATA 1, 12.0, HZ, 13.000

■ :SOURce:JITter:SMASk:TABLE:DATA? <point>

- Parameter <point>= <DECIMAL NUMERIC RESPONSE DATA>
 1 to 10 Step value: 1
- Response <freq1>,<freq2>,<amp1>
 <freq1> = XXX.X
 <amp1> = XXX.X (In the case of $0 \leq \text{value} \leq 2.000$)
 XX.XX (In the case of $2.000 < \text{value}$)
- Function Queries the jitter modulation frequency and jitter amplitude against the output signal upon Sweep mask measurement. The point frequency and amplitude specified for the <point> in the current table are returned.
- Limitation Not valid unless the jitter option is incorporated.
- Example Queries the jitter modulation frequency and jitter amplitude of point 1.
 > :SOURce:JITter:SMASk:TABLE:DATA? 1
 < 12.0,HZ,13.00

■ :SOURce:JITter:TRANSfer:TABLE:TYPE <type>

- Parameter <type>= <CHARACTER PROGRAM DATA>
 R Sets to the fixed table conforming to ITU-T.
 S Sets to the fixed table conforming to ITU-T G.736.
 U Sets to a table that can be set arbitrarily.
- Function Sets the type of table used for the Transfer measurement.
- Limitation Not valid unless the jitter option is incorporated.
 <S> can be set only when :SENSe:TELeCom:BRATe is <M2>.
 Only <U> setting is valid when :SENSe:TELeCom:BRATe is <M139>.
 This command sets the type of bit rate table concurrently selected. When setting a different type of bit rate table, change the bit rate before executing this command.
- Example Sets the bit rate 2 Mb/s table to S.
 > :SENSe:TELeCom:BRATe M2 ←Sets the bit rate to 2 Mb/s.
 > :SOURce:JITter:TRANSfer:TABLE:TYPE S ←Sets the 2M table to S.

■ :SOURce:JITter:TRANSfer:TABLE:TYPE?

- Response <type>= <CHARACTER RESPONSE DATA>
- Function Queries the type of table used for the Transfer measurement.
- Limitation Not valid unless the jitter option is incorporated.
 This query queries the type of bit rate table concurrently selected. When querying a different type of bit rate table, change the bit rate before executing this command.
- Example Queries the type of bit rate 2 Mb/s table.
 > :SENSe:TELeCom:BRATe M2 ←Sets the bit rate to 2 Mb/s.
 > :SOURce:JITter:TRANSfer:TABLE:TYPE? ←Queries the type of table.
 < S ←2M table type is returned.

■ :SOURce:JITTer:TRAnsfer:TABLE:POINt <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 10 Step value: 1
- Function Sets the number of Trasnfer measuring points.
- Limitation Not valid unless the jitter option is incorporated.
 Valid only when :SOURce:JITTer:TRAnsfer:TABLE:TYPE is <U>.
- Example Sets the number of measuring points to 5.
 > :SOURce:JITTer:TRAnsfer:TABLE:POINt 5

■ :SOURce:JITTer:TRAnsfer:TABLE:POINt?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the number of Trasnfer measuring points.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SOURce:JITTer:TRAnsfer:TABLE:POINt?
 < 5

■ :SOURce:JITTer:TRAnsfer:TABLE:DATA <point>,<freq1>,<freq2>,<amp1>

- Parameter <point>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 10 Step value : 1
 <freq1>= <DECIMAL NUMERIC PROGRAM DATA>
 1.0 to 990.0 Step value : 0.1
 <freq2>= <CHARACTER PROGRAM DATA>
 HZ,KHZ,MHZ
 <amp1>= <DECIMAL NUMERIC PROGRAM DATA>
 0.000 to 20.000 Step value : 0.001
- Function Sets the jitter modulation frequency and jitter amplitude against the output signal upon Transfer measurement.
- Limitation Not valid in the following cases:
 - ▶ When the jitter option is not incorporated.
 - ▶ When :SOURce:JITTer:TRAnsfer:TABLE:POINt is not set to <U>.
 - ▶ When the <point> value is larger than the set value of :SOURce:JITTer:TRAnsfer:TABLE:POINt.

<freq1>, <freq2>, and <amp1> may be specified regardless of other settings. If the specified value is outside the setting range, the value is changed to one that is within the range and set. (Refer to the Panel operating instructions manual for the setting range.)
- Example Sets point 1 to 12.0 Hz 13.000 UI_{P-P}.
 > :SOURce:JITTer:TRAnsfer:TABLE:DATA 1, 12.0, HZ, 13.000

■ :SOURce:JITTer:TRANsfer:TABLE:DATA? <point>

- Parameter <point>= <DECIMAL NUMERIC RESPONSE DATA>
 1 to 10 Step value: 1
- Response <freq1>,<freq2>,<amp1>
 <freq1> = XXX.X
 <amp1> = XXX.X (In the case of $0 \leq \text{value} \leq 2.000$)
 XX.XX (In the case of $2.000 < \text{value}$)
- Function Queries the jitter modulation frequency and jitter amplitude against the output signal upon Transfer measurement. The point frequency and amplitude specified for the <point> in the current table are returned.
- Limitation Not valid unless the jitter option is incorporated.
- Example Queries the jitter modulation frequency and jitter amplitude of point 1.
 > :SOURce:JITTer:TRANsfer:TABLE:DATA? 1
 < 12.0,HZ,13.00

■ :SOURce:JITTer:JFRequency:OFFSet:MIN <min>

■ :SOURce:JITTer:JFRequency:OFFSet:MAX <max>

- Parameter <min>= <DECIMAL NUMERIC PROGRAM DATA>
 -50 to +49 Step value: 1
 <max>= <DECIMAL NUMERIC PROGRAM DATA>
 -49 to +50 Step value: 1
 Sets <min> $\leq$ <max>
- Function Sets the fluctuation range of the frequency in Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid unless the offset option is incorporated.
 When <min> > <max> is set, <max> is set to the value of <min>.
- Example Sets the frequency fluctuation range to -50 ppm minimum and +50 ppm maximum.
 > :SOURce:JITTer:JFRequency:OFFSet:MIN -50 ←Sets to -50 ppm min.
 > :SOURce:JITTer:JFRequency:OFFSet:MAX +50 ←Sets to +50 ppm max.

■ :SOURce:JITter:JFRequency:OFFSet:MIN?

■ :SOURce:JITter:JFRequency:OFFSet:MAX?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the fluctuation range of the frequency in Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
Not valid unless the offset option is incorporated.
- Example > :SOURce:JITter:JFRequency:OFFSet:MIN? ←Queries the min. value.
 < -50
 > :SOURce:JITter:JFRequency:OFFSet:MAX? ←Queries the max. value.
 < 50

■ :SOURce:JITter:JFRequency:OFFSet:STEP <numeric>

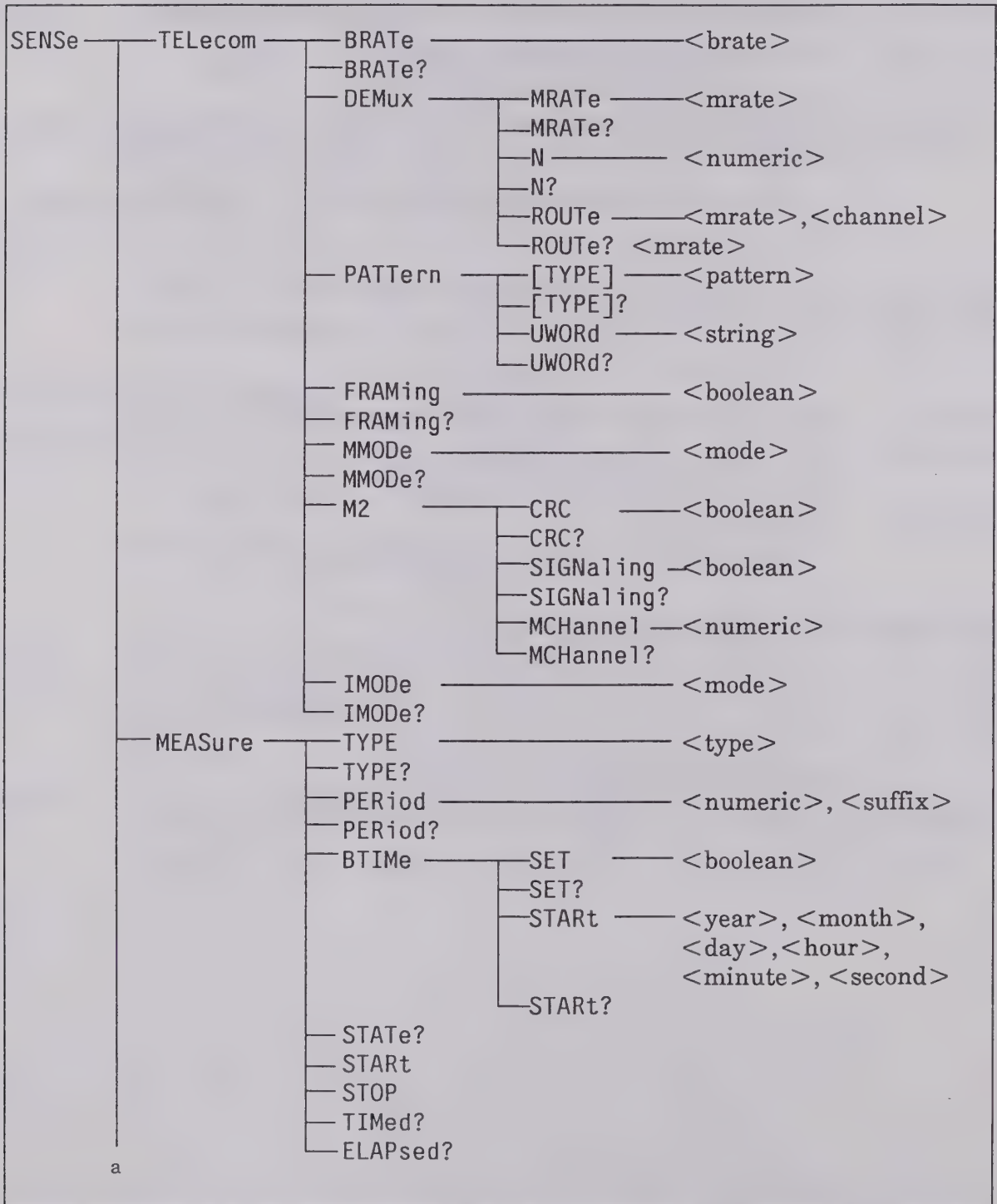
- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 100 Step value : 1
- Function Sets the variable step of frequency in Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
Not valid unless the offset option is incorporated.
- Example Sets the step to 2 ppm.
 > :SOURce:JITter:JFRequency:OFFSet:STEP 2

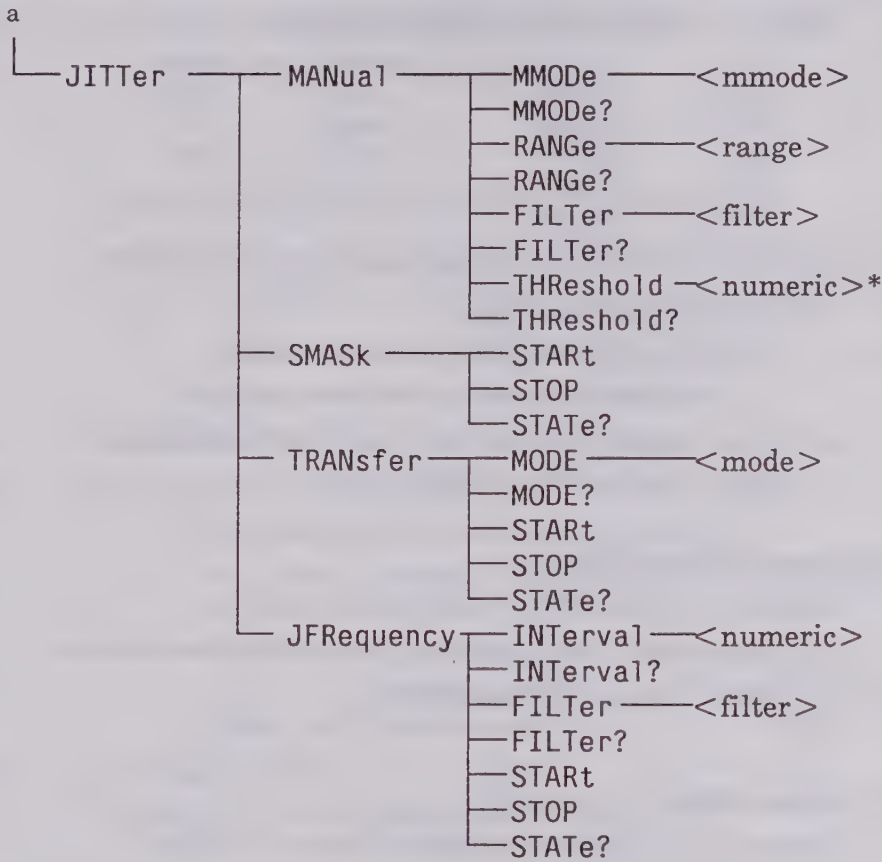
■ :SOURce:JITter:JFRequency:OFFSet:STEP?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the variable step of frequency in Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
Not valid unless the offset option is incorporated.
- Example > :SOURce:JITter:JFRequency:OFFSet:STEP?
 < 2

9.3.3 SENSE subsystem

The SENSE subsystem is used to set the measurement conditions and other settings in the receive section.





* It is possible to set a number expressed in real number format using these command. (See paragraph 4.3.2 <DECIMAL NUMERIC PROGRAM DATA>)

■ :SENSe:TELEcom:BRATe <brate>

• Parameter <brate> = <CHARACTER PROGRAM DATA>

M139	139 Mb/s
M34	34 Mb/s
M8	8 Mb/s
M2	2 Mb/s

• Function Sets the receiver signal bit rate.

• Example Sets the receiver signal bit rate to 139 Mb/s.

> :SENSe:TELEcom:BRATe M139

■ :SENSe:TELEcom:BRATe?

• Response <brate> = <CHARACTER RESPONSE DATA>

• Function Queries the receiver signal bit rate.

• Example > :SENSe:TELEcom:BRATe?
< M139

■ :SENSe:TELecom:DEMux:MRATe <mrte>

- Parameter <mrte>= <CHARACTER PROGRAM DATA>

OFF	Does not use the Demultiplexer function
M139	139 Mb/s
M34	34 Mb/s
M8	8 Mb/s
M2	2 Mb/s
K64	64 kb/s
- Function Sets the bit rate of the final tributary signal in the Demultiplexer.
- Limitation Not valid in the following cases:
 - ▶ The Demultiplexer Option is not incorporated
 - ▶ A higher order bit rate than interface speed is set
- Example Set the final tributary bit rate in the Demultiplexer to 64 kb/s.
 > :SENSe:TELecom:DEMux:MRATe K64

■ :SENSe:TELecom:DEMux:MRATe?

- Response <mrte>= <CHARACTER RESPONSE DATA>
- Function Queries the bit rate of the final tributary signal in the Demultiplexer.
- Example > :SENSe:TELecom:DEMux:MRATe?
 < K64

■ :SENSe:TELecom:DEMux:N <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1 to 31	Step value: 1
---------	---------------
- Function Sets consecutive channels of 64 kb/s.
 Receives a test signal whose N channels are consecutive from the 64 kb/s channel specified in the :SENSe:TELecom:DEMux:ROUTe command.
- Limitation Not valid unless the final tributary bit rate is specified up to 64 kb/s.
 Up to <31> can be set only when :SENSe:TELecom:M2:MCHannel is <31>.
- Example Set the consecutive channels to 5.
 > :SENSe:TELecom:DEMux:N 5

■ :SENSe:TELecom:DEMux:N?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the set value of consecutive channel at 64 kb/s.
- Example > :SENSe:TELecom:DEMux:N?
 < 5

■ :SENSe:TELeom:DEMUX:ROUTe <mrate>,<channel>

- Response <mrate>= <CHARACTER PROGRAM DATA>

M34	34 Mb/s
M8	8 Mb/s
M2	2 Mb/s
K64	64 kb/s

 <channel>= <DECIMAL NUMERIC PROGRAM DATA>

1 to 4	When <mrate> is set to M34, M8, or M2
1 to 31-N	When <mrate> is set to K64, Step value: 1 (Only 1 can be set for N=31.)

N: The value specified in :SENSe:TELeom:DEMUX:N command
- Function Sets the channel for each stage in the Demultiplexer.
- Limitation Not valid in the following cases:
 - ▶ The Demultiplexer Option is not incorporated
 - ▶ The <OFF> parameter is set in the :SENSe:TELeom:DEMUX:MRATE command
 - ▶ A higher order bit rate than interface speed is set
 - ▶ A lower order speed than the bit rate set in the :SENSe:TELeom:DEMUX:MRATE command is set
- Example Set the channel number at 64 kb/s to 12.
 > :SENSe:TELeom:DEMUX:ROUTe K64, 12

■ :SENSe:TELeom:DEMUX:ROUTe? <mrate>

- Parameter <mrate>= <CHARACTER PROGRAM DATA>
- Response <channel>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the channel which has been set in the Demultiplexer.
- Example Query the channel which has been set at 64 kb/s.
 > :SENSe:TELeom:DEMUX:ROUTe? K64
 < 12

■ :SENSe:TELEcom:PATtern[:TYPE] <pattern>

- Parameter <pattern>= <CHARACTER PROGRAM DATA>

PRBS11	PRBS2 ¹¹ -1
PRBS15	PRBS2 ¹⁵ -1
PRBS23	PRBS2 ²³ -2
UWORD2	The word data is 2 bits in length.
UWORD4	The word data is 4 bits in length.
UWORD8	The word data is 8 bits in length.
UWORD16	The word data is 16 bits in length.
- Function Sets the test pattern for the receiver signal.
- Limitation Not valid if the <ISERvice> parameter is set in the :SENSe:TELEcom:MMode command.
- Example Sets the receiver signal test pattern to the word data that is 4 bits in length.


```
> :SENSe:TELEcom:PATtern:TYPE UWORD4
or :SENSe:TELEcom:PATtern UWORD4 (TYPE can be abbreviated)
```

■ :SENSe:TELEcom:PATtern[:TYPE]?

- Response <pattern>= <CHARACTER RESPONSE DATA>
- Function Queries the receiver signal test pattern
- Example

```
> :SENSe:TELEcom:PATtern:TYPE?
or :SENSe:TELEcom:PATtern? (TYPE can be abbreviated)
<UWORD4
```

■ :SENSe:TELEcom:PATtern:UWORD <string>

- Parameter <string>= <STRING PROGRAM DATA>

"00...0" to "11 ...1"
- Function Sets the test pattern word data for the receiver signal.
- Limitation
 - ▶ The number of digits in a string corresponds to the word data length set in the :SENSe:TELEcom:PATtern[:TYPE] command.
 - ▶ Not valid if the <PRBS11, 15, 23> parameter is selected for the :SENSe:TELEcom:PATtern[:TYPE] command.
 - ▶ Not valid if the <ISERvice> parameter is set for the :SENSe:TELEcom:MMode command.
- Example Set the 4-bit word data to "1111".


```
> :SENSe:TELEcom:PATtern:UWORD "1111"
```

■ :SENSe:TELEcom:PATtern:UWORD?

- Response <string>= <STRING RESPONSE DATA>
- Function Queries the test pattern word data which has been set for the receiver signal.
- Example

```
> :SENSe:TELEcom:PATtern:UWORD?
<"1111"
```

■ :SENSe:TELeom:FRAMing <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>

OFF or 0	Without frame
ON or 1	With frame
- Function Switches the test signal frame in the receiver signal to ON or OFF.
Switches the frame of the final tributary signal to ON or OFF when the Demultiplexer (Option 02) is turned ON.
- Example Set the test signal frame in the receiver signal to ON.
 > :SENSe:TELeom:FRAMing ON

■ :SENSe:TELeom:FRAMing?

- Response <NR1 NUMERIC RESPONSE DATA>

0 or 1	
--------	--
- Function Queries the test signal frame which has been set for the receiver signal.
Queries the frame of the final tributary signal when the Demultiplexer (Option 02) is turned ON.
- Example > :SENSe:TELeom:FRAMing?
 < 1

■ :SENSe:TELeom:MMODE <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>

OSERVICE	Out-of-service measurement
ISERVICE	In-service measurement
- Function Select the measurement mode.
- Example Set the measurement mode to out-of-service.
 > :SENSe:TELeom:MMODE OSERVICE

■ :SENSe:TELeom:MMODE?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the measurement mode.
- Example > :SENSe:TELeom:MMODE?
 < OSER

■ :SENSe:TELeom:M2:CRC <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>

OFF or 0	Does not execute CRC check.
ON or 1	Executes CRC check.
- Function Switches the CRC check at 2M to ON or OFF.
- Example Set the CRC check to ON.
 > :SENSe:TELeom:M2:CRC ON

■ :SENSe:TELEcom:M2:CRC?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries the CRC check which has been set at 2 M.
- Example >:SENSe:TELEcom:M2:CRC?
 < 1

■ :SENSe:TELEcom:M2:SIGNaling <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Signaling multi-frame OFF
 ON or 1 Signaling multi-frame ON
- Function Switches the signaling multi-frame to ON or OFF.
- Limitation If the <31> parameter is set in the :SENSe:TELEcom:M2:MCHannel command, then setting to <ON> is not valid.
- Example Set the signaling multi-frame to ON.
 >:SENSe:TELEcom:M2:SIGNaling ON

■ :SENSe:TELEcom:M2:SIGNaling?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether signaling multi-frame is set or not.
- Example >:SENSe:TELEcom:M2:SIGNaling?
 < 1

■ :SENSe:TELEcom:M2:MCHannel <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 30 30 channels
 31 31 channels
- Function Sets the maximum number of channels at 2 M signal.
- Example Set the maximum number of channels to 30.
 >:SENSe:TELEcom:M2:MCHannel 30

■ :SENSe:TELEcom:M2:MCHannel?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
- Function Queries the maximum number of channels at 2 M signal.
- Example >:SENSe:TELEcom:M2:MCHannel?
 < 30

■ :SENSe:TELEcom:IMODE <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>

TERMinal	Used to connect the main output from the instrument either directly or via a specified cable.
MONItoring	Used to connect the 20 dB-attenuated main output level from the instrument either directly or via a specified cable.
- Function Sets the input signal level.
- Example Set the input level to TERMinal.
 >:SENSe:TELEcom:IMODE TERMinal

■ :SENSe:TELEcom:IMODE?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the input signal level.
- Example >:SENSe:TELEcom:IMODE?
 <TERM

■ :SENSe:MEASure:TYPE <type>

- Parameter <type>= <CHARACTER PROGRAM DATA>

MANual	Manual measurement
SINGle	Single measurement
REPeat	Repeat measurement
- Function Selects one of the measurement mode types.
- Example Set the measurement mode type to repeat measurement.
 >:SENSe:MEASure:TYPE REPeat

■ :SENSe:MEASure:TYPE?

- Response <type>= <CHARACTER RESPONSE DATA>
- Function Queries which measurement mode type has been selected.
- Example >:SENSe:MEASure:TYPE?
 <REP

■ :SENSe:MEASure:PERiod <numeric>,<suffix>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1 to 99	Step value: 1
---------	---------------

 <suffix>= <CHARACTER PROGRAM DATA>

d	day
h	hour
m	minute
s	second
- Function Sets the measurement period.

- Limitation Not valid if the <MANual> parameter is set in the :SENSe:MEASure:TYPE command.
- Example Set the measurement period to 1 hour.
> :SENSe:MEASure:PERiod 1, h

■ :SENSe:MEASure:PERiod?

- Response <numeric>, <suffix>
<numeric> = <NR1 NUMERIC RESPONSE DATA>
<suffix> = <CHARACTER RESPONSE DATA>
- Function Queries the measurement period.
- Example > :SENSe:MEASure:PERiod?
<1, h

■ :SENSe:MEASure:BTIme:SET <boolean>

- Parameter <boolean> = <BOOLEAN PROGRAM DATA>
OFF or 0 Measurement starting time setting function OFF
ON or 1 Measurement starting time setting function ON
- Function Switches the function for setting the measurement starting time to ON or OFF.
- Example Set the function for setting the measurement starting time to ON.
> :SENSe:MEASure:BTIme:SET ON

■ :SENSe:MEASure:BTIme:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
- Function Queries whether or not the function for setting the measurement starting time has been set to ON.
- Example > :SENSe:MEASure:BTIme:SET?
<1

■ :SENSe:MEASure:BTIme:START

<year>, <month>, <day>, <hour>, <minute>, <second>

- Parameter <DECIMAL NUMERIC PROGRAM DATA>

<year> =	1993 to 2092
<month> =	1 to 12
<day> =	1 to 31
<hour> =	0 to 23
<minute> =	0 to 59
<second> =	0 to 59
- Function Sets the measurement starting time.
- Limitation Not valid if the <OFF> parameter is set in the :SENSe:MEASure:BTIme:SET command.
- Example Set the measurement starting time to 10:00:00, October 10, 1993.
> :SENSe:MEASure:BTIme:START 1993, 10, 10, 0, 0

■ :SENSe:MEASure:BTIMe:START?

- Response <year>, <month>, <day>, <hour>, <minute>, <second>
 = <NR1 NUMERIC RESPONSE DATA>
- Function Queries the measurement starting time.
- Example > :SENSe:MEASure:BTIMe:START?
 < 1993,10,10,10,0,0

■ :SENSe:MEASure:STATe?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 0 Measurement end
 1 Measurement in progress
- Function Queries the measurement status.
- Example > :SENSe:MEASure:STATe?
 < 1

■ :SENSe:MEASure:START

- Parameter None
- Function Allows measurement to be started.
- Limitation Not valid while self-test is being executed.
- Example > :SENSe:MEASure:START

■ :SENSe:MEASure:STOP

- Response None
- Function Allows measurement to be stopped.
- Example > :SENSe:MEASure:STOP

■ :SENSe:MEASure:TIMed?

- Parameter <day>, <hour>, <minute>, <second>
 = <NR1 NUMERIC RESPONSE DATA>
- Function Queries the measurement time remaining.
- Example Remaining time is 15 days, 23 hours, 59 minutes, and 48 seconds
 > :SENSe:MEASure:TIMed?
 < 15,23,59,48

■ :SENSe:MEASure:ELAPsed?

- Response `<day>,<hour>,<minute>,<second>=`
 `<NR1 NUMERIC RESPONSE DATA>`
- Function Queries the measurement elapsed.
- Example Elapsed time is 5 days, 19 hours, 50 minutes, and 34 seconds
 `>:SENSe:MEASure:ELAPsed?`
 `< 5,19,50,34`

■ :SENSe:JITter:MANual:MMODE <mmode>

- Parameter `<mmode>= <CHARACTER PROGRAM DATA>`
 `INTERval02` Sets the measurement interval to 0.2 seconds.
 `INTERval1` Sets the measurement interval to 1 second.
 `INTERval10` Sets the measurement interval to 10 seconds.
 `COUPled` Sets the measurement interval so it is the same as the
 error measurement interval.
- Function Sets the jitter measurement interval.
- Limitation Not valid unless the jitter option is incorporated.
- Example Sets the measurement interval to 0.2 seconds.
 `>:SENSe:JITter:MANual:MMODE INTERval02`

■ :SENSe:JITter:MANual:MMODE?

- Response `<mmode>= <CHARACTER RESPONSE DATA>`
- Function Queries the jitter measurement interval.
- Limitation Not valid unless the jitter option is incorporated.
- Example `>:SENSe:JITter:MANual:MMODE?`
 `< INT02`

■ :SENSe:JITter:MANual:RANGe <range>

- Parameter `<range>= <CHARACTER PROGRAM DATA>`
 `UI20` Sets the range to 20UI.
 `UI2` Sets the range to 2UI.
 `AUTO` Sets the range to Auto.
- Function Sets the measurement range of the jitter amplitude.
- Limitation Not valid unless the jitter option is incorporated.
- Example Sets the range to 20UI.
 `>:SENSe:JITter:MANual:RANGe UI20`

■ :SENSe:JITter:MANual:RANGe?

- Response <range>= <CHARACTER RESPONSE DATA>
- Function Queries the measurement range of the jitter amplitude.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:SENSe:JITter:MANual:RANGe?
 <UI20

■ :SENSe:JITter:MANual:FILTer <filter>

- Parameter <filter>= <CHARACTER PROGRAM DATA>

OFF	Do not use a filter.
HP1	Use high pass filter 1.
HP2	Use high pass filter 2.
HPN2	Use high pass filter 2 (N).
LP	Use low pass filter.
LHP1	Use high pass filter 1 and low pass filter.
LHP2	Use high pass filter 2 and low pass filter.
LHON2	Use high pass filter 2 (N) and low pass filter.
- Function Sets the filter for jitter amplitude measurement.
- Limitation Not valid unless the jitter option is incorporated.
 <HPN2> and <LHPN2> can be set only when :SENSe:TELecom:BRATe is
 <M8> and <M2>.
- Example Sets the filter to high pass filter 1.
 >:SENSe:JITter:MANual:FILTer HP1

■ :SENSe:JITter:MANual:FILTer?

- Response <filter>= <CHARACTER RESPONSE DATA>
- Function Queries the filter for jitter amplitude measurement.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:SENSe:JITter:MANual:FILTer?
 <HP1

■ :SENSe:JITter:MANual:THReshold <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

0.5 to 10.0	Step value : 0.1 (Rx Range : 20UI)
0.05 to 1.00	Step value : 0.01 (Rx Range : 2UI)
- Function Sets the threshold for Hit determination.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid if :SENSe:JITter:MANual:RANGe is <AUTO>.
 Setting range and step value differ depending on the measurement range.
- Example Sets the threshold to 1.00UI_{P.P.}
 >:SENSe:JITter:MANual:THReshold 1.00

■ :SENSe:JITTer:MANual:THReshold?

- Response <numeric>= <NR2 NUMERIC RESPONSE DATA>
 XX.X (Rx Range : 20UI)
 X.XX (Rx Range : 2UI)
- Function Queries the threshold for Hit determination.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SENSe:JITTer:MANual:THReshold?
 < 1.00

■ :SENSe:JITTer:SMASk:START

- Function Starts Sweep mask measurement.
- Limitation Not valid unless the jitter option is incorporated.
 The measurement can be started only when the Jitter:Sweep mask screen is displayed.
 Not valid if another measurement is in progress.
 Restarts the measurement if Sweep mask measurement is in progress.
- Example Starts Sweep mask measurement.
 > :DISPlay:MENU "JITTer:SMASk" ←Moves to Jitter:Sweep mask screen.
 > :SENSe:JITTer:SMASk:START ←Starts measurement.

■ :SENSe:JITTer:SMASk:STOP

- Function Stops Sweep mask measurement.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid if another measurement is in progress.
- Example > :SENSe:JITTer:SMASk:STOP

■ :SENSe:JITTer:SMASk:STATe?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 Measurement end
 1 Measurement in progress
- Function Queries the status of Sweep mask measurement.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SENSe:JITTer:SMASk:STATe?
 < 1

:SENSe:JITTer:TRANsfer:MODE <mode>

- | | | |
|--------------|---|------------------------------------|
| ● Parameter | <mode>= <CHARACTER PROGRAM DATA> | |
| | MEASurement | Executes the Transfer measurement. |
| | CALibration | Executes the Calibration. |
| ● Function | Switches the measurement mode. | |
| ● Limitation | Not valid unless the jitter option is incorporated. | |
| ● Example | Sets measurement mode. | |
| | >:SENSe:JITter:TRANsfer:MODE MEASurement | |

```

■ :SENSe:JITTer:TRANsfer:MODE?

```

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the measurement mode.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SENSe:JITter:TRANsfer:MODE?
< MEAS

```
■ :SENSe:JITTer:TRANsfer:START
```

- | | |
|--------------|--|
| ● Function | Starts Transfer measurement.
Starts the measurement of the mode selected by :SENSe:JITTer:TRANSfer:MODE. |
| ● Limitation | Not valid unless the jitter option is incorporated.
The measurement can be started only when the Jitter:Transfer screen is displayed.
Not valid if another measurement is in progress.
Restarts the measurement if Transfer measurement is in progress. |
| ● Example | Starts Transfer measurement.
<pre>> :DISPlay:MENU "JITTer:TRANSfer"</pre> ← Moves to Jitter: Transfer screen.
<pre>> :SENSe:JITTer:TRANSfer:MODE MEASurement</pre> ← Sets to the measurement mode.
<pre>> :SENSe:JITTer:TRANSfer:START</pre> ← Starts measurement. |

■ :SENSe:JITTer:TRANsfer:STOP

- | | |
|--------------|---|
| • Function | Stops Transfer measurement. |
| • Limitation | Not valid unless the jitter option is incorporated.
Not valid if another measurement is in progress. |
| • Example | > :SENSe:JITTer:TRAnsfer:STOP |

■ :SENSe:JITTer:TRANsfer:STATe?

- Response <NR1 NUMERIC RESPONSE DATA>

0	Measurement end
1	Measurement in progress
- Function Queries the status of Transfer measurement.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SENSe:JITTer:TRANsfer:STATe?
< 1

■ :SENSe:JITTer:JFRequency:INTerval <numeric>

- Parameter <numeric>= <DECIMAL PROGRAM DATA>
 1 to 60 Step value:1
- Function Sets the measurement time per point in Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid unless the offset option is incorporated.
- Example Sets the measurement time to 1 sec.
> :SENSe:JITTer:JFRequency:INTerval 1

■ :SENSe:JITTer:JFRequency:INTerval?

- Response <NR1 NUMERIC RESPONSE DATA>
- Function Queries the measurement time per point.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid unless the offset option is incorporated.
- Example > :SENSe:JITTer:JFRequency:INTerval?
< 1

■ :SENSe:JITTer:JFRequency:FILTer <filter>

- Parameter <filter>= <CHARACTER PROGRAM DATA>

OFF	Do not use a filter.
HP1	Use high pass filter 1.
HP2	Use high pass filter 2.
HPN2	Use high pass filter 2 (N).
LP	Use low pass filter.
LHP1	Use high pass filter 1 and low pass filter.
LHP2	Use high pass filter 2 and low pass filter.
LHON2	Use high pass filter 2 (N) and low pass filter.

- Function Sets the filter for amplitude measurement in Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
Not valid unless the offset option is incorporated.
<HPN2> and <LHPN2> can be set only when :SENSe:TELecom:BRATe is <M8> and <M2>.
- Example Sets the filter to high pass filter 1.
> :SENSe:JITTer:JFRequency:FILTer HP1

■ :SENSe:JITTer:JFRequency:FILTer?

- Response <filter>= <CHARACTER RESPONSE DATA>
- Function Queries the filter for magnitude measurement in Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
Not valid unless the offset option is incorporated.
- Example > :SENSe:JITTer:JFRequency:FILTer?
< HP1

■ :SENSe:JITTer:JFRequency:START

- Function Starts Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is not incorporated.
Not valid unless the offset option is not incorporated.
The measurement can be started only when the Jitter:Jitt/Freq screen is displayed.
.. Not valid if another measurement is in progress.
Restarts the measurement if Jitt/Freq measurement is in progress.
- Example Starts Jitt/Freq measurement.
> :DISPlay:MENU "JITTer:JFRequency" ←Moves to Jitter:Jitt/Freq screen.
> :SENSe:JITTer:JFRequency:START ←Starts measurement.

■ :SENSe:JITTer:JFRequency:STOP

- Function Stops Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
Not valid unless the offset option is incorporated.
Not valid if another measurement is in progress.
- Example > :SENSe:JITTer:JFRequency:STOP

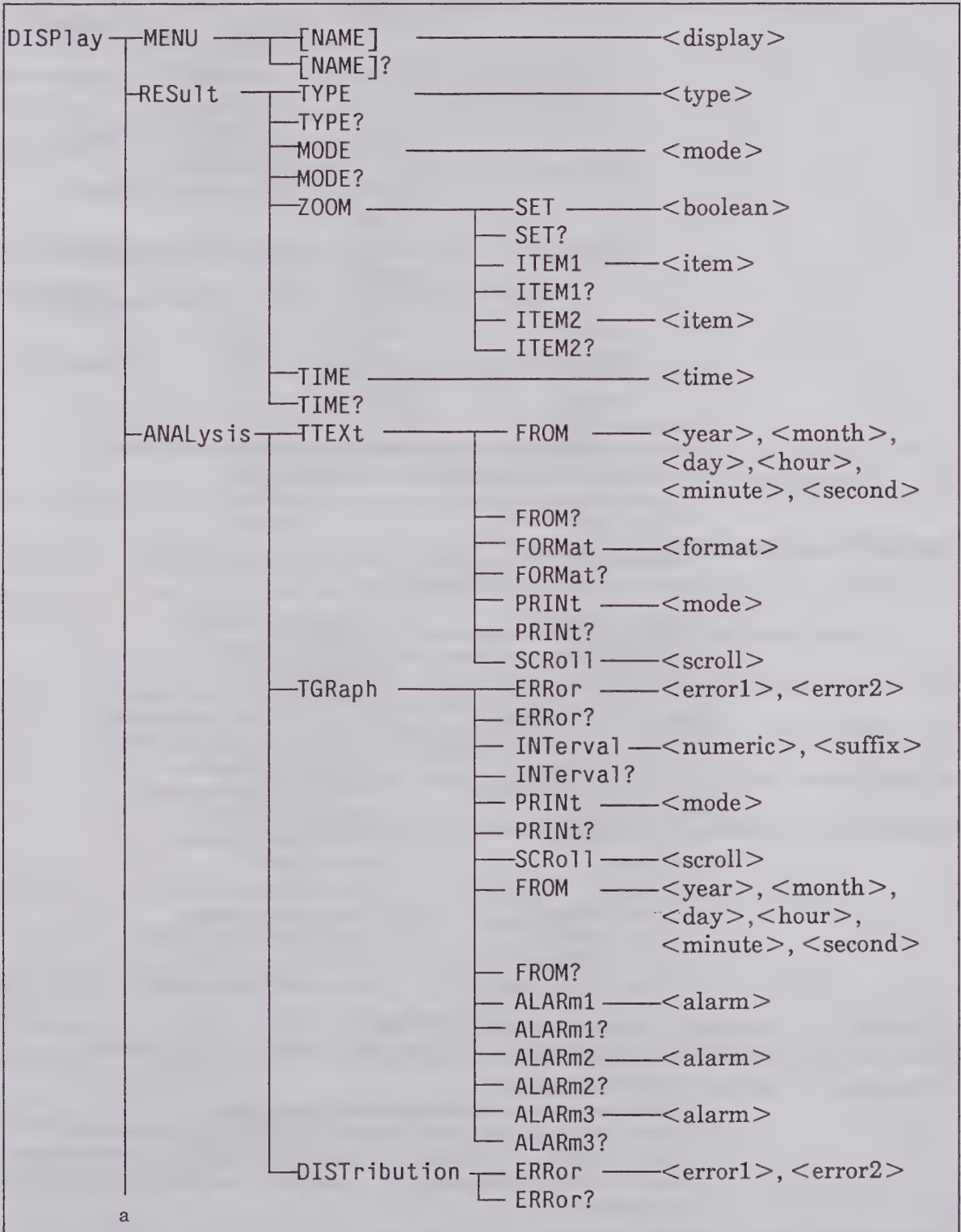
■ :SENSe:JITTer:JFRequency:STAtE?

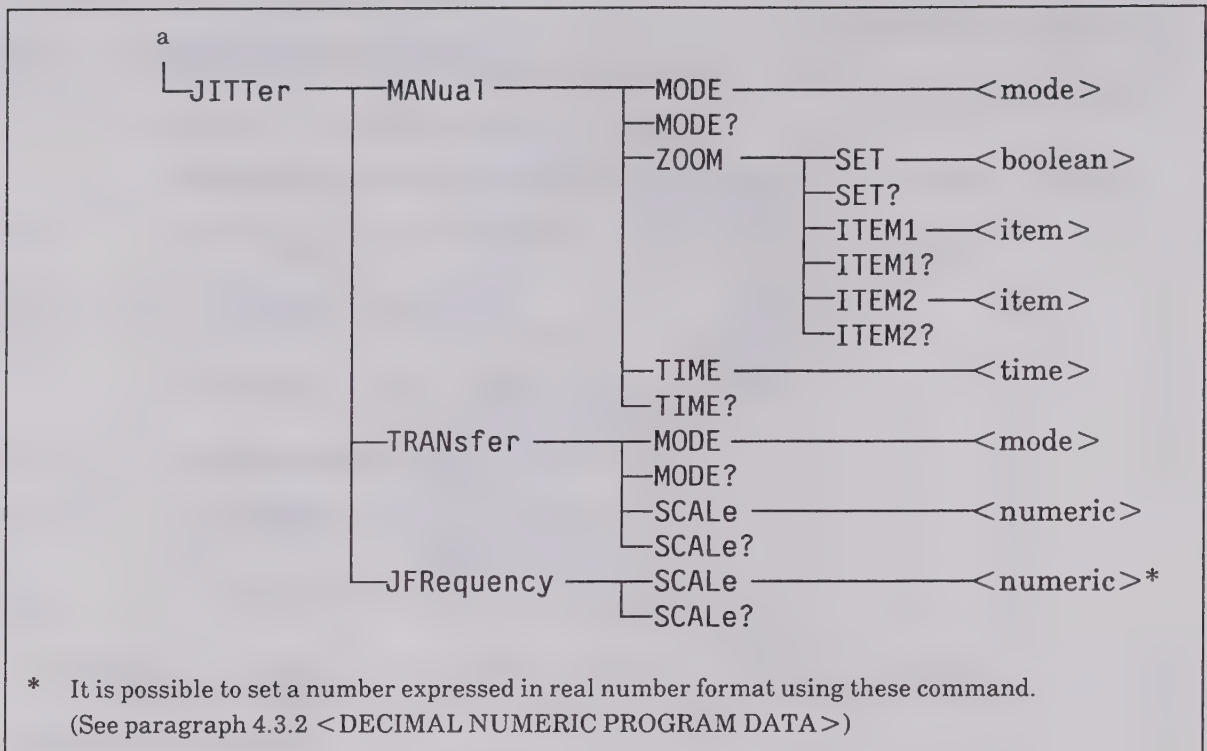
- Response <NR1 NUMERIC RESPONSE DATA>

0	Measurement end
1	Measurement in progress
- Function Queries the status of Jitt/Freq measurement.
- Limitation Not valid unless the jitter option is incorporated.
Not valid unless the offset option is incorporated.
- Example > :SENSe:JITTer:JFRequency:STAtE?
< 1

9.3.4 DISPlay subsystem

The DISPlay subsystem is used for various settings when displaying the Result screen, the Analysis screen, and other screens.





■ :DISPlay:MENU[:NAME] <display>

- Parameter <display>= <STRING PROGRAM DATA>

"CONDtn:TX&RX"	Condtn:Tx&Rx screen
"CONDtn:TX"	Condtn:Tx screen
"CONDtn:RX"	Condtn:Rx screen
"ANALysis:TTExt"	Analysis:Trace Text screen
"ANALysis:TGRaph"	Analysis:Trace Graph screen
"ANALysis:DISTribution"	Analysis:Distribution screen
"SETup:MEASure"	Setup:Measure screen
"SETup:PERformance"	Setup:Performance screen
"SETup:PRINt"	Setup:Print screen
"SETup:MEMory"	Setup:Memory screen
"SYSTem"	System screen
"SELFtest"	Self-test screen
"JITter:MANUal"	Jitter:Manual screen
"JITter:SMASk"	Jitter:Sweep mask screen
"JITter:TRANSfer"	Jitter:Transfer screen
"JITter:JFRequency"	Jitter:Jitt/Freq screen

- Function Switches the screens listed above for display.
Use the :DISPlay:RESult:TYPE command to switch the Result screen.
- Limitation Not valid in the following cases:
 - ▶ The Self-test screen is specified during measurement
 - ▶ Screens other than the Self-test screen are specified during self-test
 - ▶ When "<JITter:..>" is specified when the jitter option is not incorporated.
 - ▶ When "<JITter:JFRequency>" is specified when the offset option is not incorporated.

- Example Switch the display screen to the Analysis:Trace Graph screen.
`>:DISPlay:MENU:NAME "ANALysis:TGRaph"`
or `:DISPlay:MENU "ANALysis:TGRaph"` (NAME can be abbreviated)

■ :DISPlay:MENU[:NAME]?

- Response `<display>=<STRING RESPONSE DATA>`
- Function Queries the screen name which has been switched.
- Example `>:DISPlay:MENU:NAME?`
or `:DISPlay:MENU? (NAME can be abbreviated)`
`<"ANAL:TGR"`

■ :DISPlay:RESult:TYPE <type>

- Parameter `<type>=<STRING PROGRAM DATA>`

<code>"ERRor:BIT"</code>	Bit error
<code>"ERRor:CODE"</code>	Code error
<code>"ERRor:FRAMe:M139"</code>	139M frame error
<code>"ERRor:FRAMe:M34"</code>	34M frame error
<code>"ERRor:FRAMe:M8"</code>	8M frame error
<code>"ERRor:FRAMe:M2"</code>	2M frame error
<code>"ERRor:CRC"</code>	CRC-4 error
<code>"ALARm1"</code>	Alarm group 1
<code>"ALARm2"</code>	Alarm group 2
- Function Sets the display group and display error of the Result screen.
- Limitation Not valid in the following cases:
 - ▶ The `<ERRor:BIT>` parameter is set during in-service measurement
 - ▶ A higher order frame error than interface is set
 - ▶ The `<ERRor:CRC>` parameter is set with the `<OFF>` parameter set in the `:SENSe:TELeCom:M2:CRC` command
 - ▶ A lower order frame error than the final tributary signal is set in the Demultiplexer (Option 02)
 - ▶ The final tributary frame error of the Demultiplexer is set with the frame set to `<OFF>`
 - ▶ The `<ERRor:CRC>` parameter is set when the option is not demultiplexed until 2M
- Example Display bit error.
`>:DISPlay:RESult:TYPE "ERRor:BIT"`

■ :DISPlay:RESult:TYPE?

- Response `<type>=<STRING RESPONSE DATA>`
- Function Queries which error or group has been set for the display error or display group on the Result screen.
- Example `>:DISPlay:RESult:TYPE?`
`<"ERRor:BIT"`

■ :DISPlay:RESult:MODE <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>
 CURRENT Current measured result
 LAST Last measured result
- Function Selects the Result screen display mode.
- Example Display the current measured result.
 >:DISPlay:RESult:MODE CURRent

■ :DISPLay:RESult:MODE?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the Result screen display mode.
- Example >:DISPLay:RESult:MODE?
 < CURR

■ :DISPLay:RESult:ZOOM:SET <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Zoom-up display OFF
 ON or 1 Zoom-up display ON
- Function Switches the zoom-up display to ON or OFF.
- Example Switch the zoom to ON.
 >:DISPLay:RESult:ZOOM:SET ON

■ :DISPlay:RESult:ZOOM:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether zoom-up display is set or not
- Example >:DISPlay:RESult:ZOOM:SET?
 < 1

■ :DISPLay:RESuLt:ZOOM:ITEM1 <item>

- Parameter <item>= <STRING PROGRAM DATA>

"EC"	Error count	"M2100:ES"	M.2100 ES
"ER"	Error rate	"M2100:SES"	M.2100 SES
"ES"	Error second	"LOF:M139"	139 Mb/s frame loss
"Power"	Power fail	"LOF:M34"	34 Mb/s frame loss
"LOS"	Signal loss	"LOF:M8"	8 Mb/s frame loss
"PAtTern"	Pattern sync	"LOF:M2"	2 Mb/s frame loss
"FREQuency"	Frequency offset	"LOF:MF"	Multi-frame loss
"G821:ES"	G.821 %ES	"AIS:M139"	139 Mb/s AIS
"G821:SES"	G.821 %SES	"AIS:M34"	34 Mb/s AIS
"G821:DM"	G.821 %DM	"AIS:M8"	8 Mb/s AIS
"G821:US"	G.821 %US	"AIS:M2"	2 Mb/s AIS
"G826:ESR"	G.826 ESR	"RA:M139"	139 Mb/s remote alarm
"G826:SESR"	G.826 SESR	"RA:M34"	34 Mb/s remote alarm
"G826:US"	G.826 US	"RA:M8"	8 Mb/s remote alarm
"G826:BBER"	G.826 BBER	"RA:M2"	2 Mb/s remote alarm
		"RA:MF"	Multi-frame remote alarm

- Function Selects the type (item1) of zoom-up display.
- Limitation
 - ▶ The item to be displayed for zoom-up must be selected from the items displayed on the screen set in the :DISPLay:RESuLt:TYPE command.
 - ▶ Not valid if the <OFF> parameter is set in the :DISPLay:RESuLt:ZOOM:SET command.
- Example Select the error count for zoom display.
>:DISPLay:RESuLt:ZOOM:ITEM1 "EC"

■ :DISPLay:RESuLt:ZOOM:ITEM1?

- Response <item>= <STRING RESPONSE DATA>
- Function Queries which type (item1) has been selected for zoom-up display.
- Example >:DISPLay:RESuLt:ZOOM:ITEM1?
< "EC"

■ :DISPLay:RESuLt:ZOOM:ITEM2 <item>

- Parameter <item>= <STRING PROGRAM DATA>

-Same as the parameter of the :DISPLay:RESuLt:ZOOM:ITEM1 command

- Function Selects the type (item2) of zoom-up display.
- Limitation
 - ▶ The item to be displayed for zoom-up must be selected from the items displayed on the screen set in the :DISPLay:RESuLt:TYPE command.
 - ▶ Not valid if the <OFF> parameter is set in the :DISPLay:RESuLt:ZOOM:SET command.
- Example Select the error rate for zoom display.
>:DISPLay:RESuLt:ZOOM:ITEM2 "ER"

■ :DISPlay:RESult:ZOOM:ITEM2?

- Response <item>= <STRING RESPONSE DATA>
- Function Queries which type (item2) has been selected for zoom-up display.
- Example > :DISPlay:RESult:ZOOM:ITEM2?
 < "ER"

■ :DISPlay:RESult:TIME <time>

- Parameter <item>= <CHARACTER PROGRAM DATA>

ELAPsed	Period of time elapsed
TIMed	Period of time remaining
DTIME	Current time
- Function Selects either a period of time or the current time for measurement displayed on the Result screen.
- Example Display the period of time remaining.
 > :DISPlay:RESult:TIME TIMed

■ :DISPlay:RESult:TIME?

- Response <time>= <CHARACTER RESPONSE DATA>
- Function Queries the period of time or the current time which has been displayed on the Result screen.
- Example > :DISPlay:RESult:TIME?
 < TIM

■ :DISPlay:ANALysis:TTExt:FROM

<year>,<month>,<day>,<hour>,<minute>,<second>

- Parameter <DECIMAL NUMERIC PROGRAM DATA>

<year> =	1993 to 2092
<month> =	1 to 12
<day> =	1 to 31
<hour> =	0 to 23
<minute> =	0 to 59
<second> =	0 to 59
- Function Specifies the time to start displaying the Analysis:Trace Text screen.
- Example Start displaying Analysis:Trace Text screen at 12:00:00, October 21, 1993.
 > :DISPlay:ANALysis:TTExt:FROM 1993,10,21,12,0,0

■ :DISPlay:ANALysis:TTExt:FROM?

- Response <year>,<month>,<day>,<hour>,<minute>,<second>=
 <NR1 NUMERIC RESPONSE DATA>
- Function Queries the time at which the Analysis:Trace Text screen will be displayed.
- Example > :DISPlay:ANALysis:TTExt:FROM?
 < 1993,10,21,12,0,0

■ :DISPlay:ANALysis:TTEXT:FORMat <format>

- Parameter <format>= <CHARACTER PROGRAM DATA>

EC	Error count
ER	Error rate

- Function Sets the display item on the Analysis:Trace Text screen.
- Example Display error count.
 >:DISPlay:ANALysis:TTEXT:FORMat EC

■ :DISPlay:ANALysis:TTEXT:FORMat?

- Response <format>= <CHARACTER RESPONSE DATA>
- Function Queries the display item which has been selected for the Analysis:Trace Text screen.
- Example >:DISPlay:ANALysis:TTEXT:FORMat?
 < EC

■ :DISPlay:ANALysis:TTEXT:PRINt <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>

ALL	All ranges in which log data is included
DISPlay	A range displayed on the screen
AFTer	The log data from and including that displayed on the screen
BEFOre	The log data up to and including that displayed on the screen

- Function Sets the range for printing out the Analysis:Trace Text screen.
- Example Print out the range displayed on the screen.
 >:DISPlay:ANALysis:TTEXT:PRINt DISPlay

■ :DISPlay:ANALysis:TTEXT:PRINt?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the print range which has been set for the Analysis:Trace Text screen.
- Example >:DISPlay:ANALysis:TTEXT:PRINt?
 < DISP

■ :DISPlay:ANALysis:TTEXT:SCRoll <scroll>

- Parameter <scroll>= <CHARACTER PROGRAM DATA>

UP	Scrolls the screen upward
DOWN	Scrolls the screen downward
TOP	Moves the screen to the beginning of the log data
BOTTOM	Moves the screen to the end of the log data

- Function Scrolls the Analysis:Trace Text screen.
- Example Scroll the screen downward.
 >:DISPlay:ANALysis:TTEXT:SCRoll DOWN

■ :DISPlay:ANALysis:TGRaph:ERRor <error1>,<error2>

- Parameter <error1>= <STRING PROGRAM DATA>

"BIT"	Bit error
"CODE"	Code error
"CRC"	CRC error
"FRAME:M139"	139M frame error
"FRAME:M34"	34M frame error
"FRAME:M8"	8M frame error
"FRAME:M2"	2M frame error

- <error2>= <CHARACTER PROGRAM DATA>

EC	Error count
ER	Error rate

- Function Sets the display error and item of the Analysis:Trace Graph screen.

- Limitation Not valid in the following cases:

- ▶ The <BIT> parameter is set for <error1> during in-service measurement.
- ▶ A higher order frame error than interface is set.
- ▶ The <CRC> parameter is set for <error1> with the <OFF> parameter set in the :SENSe:TELEcom:M2:CRC command.
- ▶ A lower order frame error than the final tributary signal is set in the Demultiplexer (Option 02).
- ▶ The final tributary frame error of the Demultiplexer is set with the frame set to <OFF>.
- ▶ The <CRC> parameter is set for <error1> when the option is not demultiplexed until 2M.

- Example Display the error rate for code error.

```
>:DISPlay:ANALysis:TGRaph:ERRor "CODE", ER
```

■ :DISPlay:ANALysis:TGRaph:ERRor?

- Response <error1>, <error2>

```
<error1>= <STRING RESPONSE DATA>
```

```
<error2>= <CHARACTER RESPONSE DATA>
```

- Function Queries any one of the display errors or display items which have been selected on the Analysis:Trace Graph screen.

- Example >:DISPlay:ANALysis:TGRaph:ERRor?

```
<"CODE", ER
```

■ :DISPlay:ANALysis:TGRaph:INTerval <numeric>,<suffix>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 10
 <suffix>= <CHARACTER PROGRAM DATA>
 d day
 h hour
 m minute
 s second
- Function Sets the width of the bar graph to display on the Analysis:Trace Graph screen.
- Limitation Any setting other than <1,s>, <10,s>, <1,m>, <10,m>, <1,h>, <6,h>, <1,d>, and <4,d> is invalid.
- Example Set the width of the bar graph to one minute.
 >:DISPlay:ANALysis:TGRaph:INTerval 1, m

■ :DISPlay:ANALysis:TGRaph:INTerval?

- Response <numeric>, <suffix>
 <numeric>= <NR1 NUMERIC RESPONSE DATA>
 <suffix> = <CHARACTER RESPONSE DATA>
- Function Queries the width of the bar graph which has been displayed on the Analysis:Trace Graph screen.
- Example >:DISPlay:ANALysis:TGRaph:INTerval?
 <1, m

■ :DISPlay:ANALysis:TGRaph:PRINT <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>
 ALL All ranges in which log data are included
 DISPlay A range displayed on the screen
 AFTer The log data from and including that displayed on the screen
 BEFore The log data up to and including that displayed on the screen
- Function Sets the range for printing out the Analysis:Trace Text screen.
- Example Print out all ranges in which log data is included.
 >:DISPlay:ANALysis:TGRaph:PRINT ALL

■ :DISPlay:ANALysis:TGRaph:PRINT?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the print range which has been set for the Analysis:Trace Graph screen.
- Example >:DISPlay:ANALysis:TGRaph:PRINT?
 <ALL

■ :DISPlay:ANALysis:TGRaph:SCRoIl <scroll>

- Parameter <scroll>= <CHARACTER PROGRAM DATA>

LEFT	Scrolls the screen to the left
RIGHT	Scrolls the screen to the right
TOP	Moves the screen to the beginning of the log data
BOTTOM	Moves the screen to the end of the log data

- **Function** Scrolls the Analysis:Trace Graph screen.
- **Example** Move the screen to the beginning of the log data
>:DISPlay:ANALysis:TGRaph:SCRo11 TOP

■ :DISPLay:ANALysis:TGRaph:FROM

<year>,<month>,<day>,<hour>,<minute>,<second>

- **Parameter** <DECIMAL NUMERIC PROGRAM DATA>

<year> =	1993 to 2092
<month> =	1 to 12
<day> =	1 to 31
<hour> =	0 to 23
<minute> =	0 to 59
<second> =	0 to 59
- **Function** Specifies the time to start displaying the Analysis:Trade Graph screen.
- **Example** Start displaying the Analysis:Trace Graph screen at 11:30:00, July 28, 1993.
>:DISPLay:ANALysis:TGRaph:FROM 1993,7,28,11,30,0

■ :DISPLay:ANALysis:TGRaph:FROM?

- **Response** <year>,<month>,<day>,<hour>,<minute>,<second>
= <NR1 NUMERIC RESPONSE DATA>
- **Function** Queries the time at which the Analysis:Trace Graph screen will be displayed.
- **Example** **>:DISPLay:ANALysis:TGRaph:FROM?**
<1993,7,28,11,30,0

■ :DISPLay:ANALysis:TGRaph:ALARm1 <a1arm>

- **Parameter** <a1arm>= <STRING PROGRAM DATA>

"ALL"	All alarm	"AIS:M139"	139M AIS
"POWer"	Power fail	"AIS:M34"	34M AIS
"LOS"	Signal loss	"AIS:M8"	8M AIS
"PATtern"	Pattern sync loss	"AIS:M2"	2M AIS
"LOF:M139"	139M frame loss	"RA:M139"	139M remote alarm
"LOF:M34"	34M frame loss	"RA:M34"	34M remote alarm
"LOF:M8"	8M frame loss	"RA:M8"	8M remote alarm
"LOF:M2"	2M frame loss	"RA:M2"	2m remote alarm
"LOF:MF"	Multi-frame loss	"RA:MF"	Multi-frame remote alarm
- **Function** Sets the display alarm of the Analysis:Trace Graph screen.
- **Limitation** Not valid in the following cases:
 - ▶ A higher order alarm than interface is set.
 - ▶ The <RA:MF> or <LOF:MF> parameter is set with the <OFF> parameter set in the :SENSe:TELEcom:M2:SIGNALing command.

- ▶ A lower order frame error than the final tributary signal is set in the Demultiplexer (Option 02).
- ▶ The final tributary RA or LOF of the Demultiplexer is set with the frame set to <OFF>.

● Example Display all alarms
 >:DISPlay:ANALysis:TGRaph:ALARm1 "ALL"

■ :DISPlay:ANALysis:TGRaph:ALARm1?

● Response <alarm>= <STRING RESPONSE DATA>
 ● Function Queries the display alarm which has been selected on the Analysis:Trace Graph screen.
 ● Example >:DISPlay:ANALysis:TGRaph:ALARm1?
 <"ALL"

■ :DISPlay:ANALysis:TGRaph:ALARm2 <alarm>

● Parameter <alarm>= <STRING PROGRAM DATA>
 Same as the parameter of the :DISPlay:ANALysis:TGRaph:ALARm1.
 ● Function Sets the display alarm of the Analysis:Trace Graph screen.
 ● Limitation Same as the limitation of the :DISPlay:ANALysis:TGRaph:ALARm1
 ● Example Display 130M AIS.
 >:DISPlay:ANALysis:TGRaph:ALARm2 "AIS:M139"

■ :DISPlay:ANALysis:TGRaph:ALARm2?

● Response <alarm>= <STRING RESPONSE DATA>
 ● Function Queries the display alarm which has been selected on the Analysis:Trace Graph screen.
 ● Example >:DISPlay:ANALysis:TGRaph:ALARm2?
 <"AIS:M139"

■ :DISPlay:ANALysis:TGRaph:ALARm3 <alarm>

● Parameter <alarm>= <STRING PROGRAM DATA>
 Same as the parameter of the :DISPlay:ANALysis:TGRaph:ALARm1.
 ● Function Sets the display alarm of the :ANALysis:Trace:Graph screen.
 ● Limitation Same as the limitation of the :DISPlay:ANALysis:TGRaph:ALARm1
 ● Example Display multi-frame remote alarm.
 >:DISPlay:ANALysis:TGRaph:ALARm3 "RA:MF"

■ :DISPlay:ANALysis:TGRaph:ALARm3?

● Response <alarm>= <STRING RESPONSE DATA>
 ● Function Queries the display alarm which has been selected on the :DISPlay:ANALysis:TGRaph:ALARm1.
 ● Example >:DISPlay:ANALysis:TGRaph:ALARm3?
 <"RA:MF"

■ :DISPlay:ANALysis:DISTribution:ERRor <error1>,<error2>

- Parameter <error1>= <STRING PROGRAM DATA>

"BIT"	Bit error
"CODE"	Code error
"CRC"	CRC error
"FRAME:M139"	139M frame error
"FRAME:M34"	34M frame error
"FRAME:M8"	8M frame error
"FRAME:M2"	2M frame error

<error2>= <CHARACTER PROGRAM DATA>

EC	Error count
ER	Error rate

- Function Sets the display error and display item of the Analysis:Distribution screen.
- Limitation Not valid i the following cases:
 - ▶ The <BIT> parameter is set for <error1> during in-service measurement.
 - ▶ A higher order frame error than interface is set.
 - ▶ The <CRC> parameter is set for <error1> with the <OFF> parameter set in the :SENSe:TELEcom:M2:CRC command.
 - ▶ A lower order frame error than the final tributary signal is set in the Demultiplexer (Option 02).
 - ▶ The final tributary frame error of the Demultiplexer is set with the frame set to <OFF>.
 - ▶ The <CRC> parameter is set for the <error1> when the option is not demultiplexed until 2M.
- Example Display the bit error rate.
>:DISPlay:ANALysis:DISTribution:ERRor "BIT", ER

■ :DISPlay:ANALysis:DISTribution:ERRor?

- Response <error1>, <error2>
<error1>= <STRING RESPONSE DATA>
<error2>= <CHARACTER RESPONSE DATA>
- Function Queries any one of the display errors or display items which have been selected on the Analysis:Distribution screen.
- Example >:DISPlay:ANALysis:DISTribution:ERRor?
< "BIT", ER

■ :DISPlay:JITTer:MANual:MODE <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>

CURRent	Displays the current measurement result.
LAST	Displays the immediately preceding measurement result.
- Function Sets the jitter measurement display method in the Jitter:Manual screen.
- Limitation Not valid unless the jitter option is incorporated.

- Example Displays the current measurement result.
 >:DISPlay:JITTer:MANual:MODE CURRent

■ :DISPlay:JITTer:MANual:MODE?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the jitter measurement display method in the Jitter:Manual screen.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:DISPlay:JITTer:MANual:MODE?
 <CURR

■ :DISPlay:JITTer:MANual:ZOOM:SET <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Cancels the zoom display.
 ON or 1 Specifies the zoom display.
- Function Sets whether or not to zoom display the jitter measurement result in the Jitter:Manual screen.
- Limitation Not valid unless the jitter option is incorporated.
- Example Specifies the zoom display.
 >:DISPlay:JITTer:MANual:ZOOM:SET ON

■ :DISPlay:JITTer:MANual:ZOOM:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Sets whether or not to zoom display the jitter measurement result in the Jitter:Manual screen.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:DISPlay:JITTer:MANual:ZOOM:SET?
 <1

■ :DISPlay:JITTer:MANual:ZOOM:ITEM1 <item>

- Parameter ...<item>= <STRING PROGRAM DATA>
 "TX:AMPLitude" Tx amplitude "HIT:COUNt" Hit count
 "RX:AMPLitude" Rx amplitude "HIT:SECONDS" Hit seconds
 "RX:PPEak" Rx + peak "HIT:FS" %Hit FS
 "RX:MPEak" Rx - peak
- Function Selects the displayed items (upper row in the screen) when the Jitter:Manual screen is in zoom display status.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid if :DISPlay:JITTer:MANual:ZOOM:SET is set to <OFF>.
 The settings of <"HIT:COUNt">, <"HIT:SECONDS"> and <"HIT:FS"> are invalid when :SENSe:JITTer:MANual:RANGE is set to <AUTO>.

- Example Zoom displays the Rx amplitude in the upper row.
 >:DISPlay:JITTer:MANual:ZOOM:ITEM1 "RX:AMPLitude"

■ :DISPlay:JITTer:MANual:ZOOM:ITEM1?

- Response <item>= <STRING RESPONSE DATA>
- Function Queries the displayed items (upper row in the screen) when the Jitter:Manual screen is in zoom display status.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:DISPlay:JITTer:MANual:ZOOM:ITEM1?
 < "RX:AMPL "

■ :DISPlay:JITTer:MANual:ZOOM:ITEM2 <item>

- Parameter <item>= <STRING PROGRAM DATA>
 The same as the parameter :DISPlay:JITTer:MANual:ZOOM:ITEM1
- Function Selects the displayed items (lower row in the screen) when the Jitter:Manual screen is in zoom display status.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid if :DISPlay:JITTer:MANual:ZOOM:SET is set to <OFF>.
 The settings of <"HIT:COUNT">, <"HIT:SECONDS"> and <"HIT:FS"> are invalid when :SENSe:JITTer:MANual:RANGe is set to <AUTO>.
- Example Zoom displays the Hit count in the lower row.
 >:DISPlay:JITTer:MANual:ZOOM:ITEM2 "HIT:COUNT"

■ :DISPlay:JITTer:MANual:ZOOM:ITEM2?

- Response <item>= <STRING RESPONSE DATA>
- Function Queries the displayed items (lower row in the screen) when the Jitter:Manual screen is in zoom display status.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:DISPlay:JITTer:MANual:ZOOM:ITEM2?
 < "HIT:COUNT"

■ :DISPlay:JITTer:MANual:TIME <time>

- Parameter <time>= <CHARACTER PROGRAM DATA>
 ELAPsed Elapsed time
 TIMed Remaining time
 DTIMed Current time
- Function Sets the measurement time display in the Jitter:Manual screen.
- Limitation Not valid unless the jitter option is incorporated.
 The setting of <TIMed> is invalid when :SENSe:MEASure:TYPE is set to <MANual>.

- Example Displays the elapsed time from the time the measurement was started.
 >:DISPlay:JITTer:MANual:TIME ELAPsed

■ :DISPlay:JITTer:MANual:TIME?

- Response <time>= <CHARACTER RESPONSE DATA>
- Function Queries the measurement time display in the Jitter:Manual screen.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:DISPlay:JITTer:MANual:TIME?
 < ELAP

■ :DISPlay:JITTer:TRANSfer:MODE <mode>

- Parameter <mode>= <CHARACTER PROGRAM DATA>

MGRaph	Displays the Transfer measurement data in a graph.
MTABle	Displays the Transfer measurement data in numbers.
CTABle	Displays the Calibration values in numbers.
- Function Selects the contents displayed in the Jitter/Transfer screen.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid if :SENSe:JITTer:TRANSfer:MODE is set to <CALibration>.
- Example Displays the Transfer measurement data in a graph.
 >:DISPlay:JITTer:TRANSfer:MODE MGRaph

■ :DISPlay:JITTer:TRANSfer:MODE?

- Response <mode>= <CHARACTER RESPONSE DATA>
- Function Queries the contents displayed in the Jitter:Transfer screen.
- Limitation Not valid unless the jitter option is incorporated.
- Example >:DISPlay:JITTer:TRANSfer:MODE?
 < MGR

■ :DISPlay:JITTer:TRANSfer:SCALE <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

10	10 dB per scale
1	1 dB per scale
- Function Sets the scale displayed in the Jitter:Transfer screen.
- Limitation Not valid unless the jitter option is incorporated.
 Valid only when :DISPlay:JITTer:TRANSfer:MODE is <MGRaph>.
- Example Sets the scale in the graph to 10 dB per scale.
 >:DISPlay:JITTer:TRANSfer:SCALE 10

■ :DISPlay:JITter:TRANsfer:SCALE?

- Response <numeric> = <NR1 NUMERIC RESPONSE DATA>
- Function Queries the scale of the graph displayed in the Jitter:Transfer screen.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :DISPlay:JITter:TRANsfer:SCALE?
 < 10

■ :DISPlay:JITter:JFRequency:SCALE <numeric>

- Parameter <numeric> = <DECIMAL NUMERIC PROGRAM DATA>

2.0	2.0UI _{P.P} per scale
1.0	1.0UI _{P.P} per scale
0.2	0.2UI _{P.P} per scale
0.1	0.1UI _{P.P} per scale
- Function Sets the scale of the graph displayed in the Jitter:Jitt/Freq screen.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid unless the offset option is incorporated.
- Example Sets the scale in the graph to 2.0UI_{P.P} per scale.
 > :DISPlay:JITter:JFRequency:SCALE 2.0

■ :DISPlay:JITter:JFRequency:SCALE?

- Response <numeric> = <NR2 NUMERIC RESPONSE DATA>
- Function Queries the scale of the graph displayed in the Jitter:Jitt/Freq screen.
- Limitation Not valid unless the jitter option is incorporated.
 Not valid unless the offset option is incorporated.
- Example > :DISPlay:JITter:JFRequency:SCALE?
 < 2.0

9.3.5 ROUTe subsystem

The ROUTe subsystem is used to set the connectors for signal routing.

```

ROUTe ——— INPut ——— SElect <port>
                      |
                      SElect?
  
```

■ :ROUTe:INPut:SElect <port>

- Parameter <port>= <CHARACTER PROGRAM DATA>

PORT1	Unbalanced connector
PORT2	Balanced connector
- Function Selects the signal input connector when the receiver signal is 2M.
- Example Set the input connector to unbalanced connector


```
> :ROUTe:INPut:SElect PORT1
```

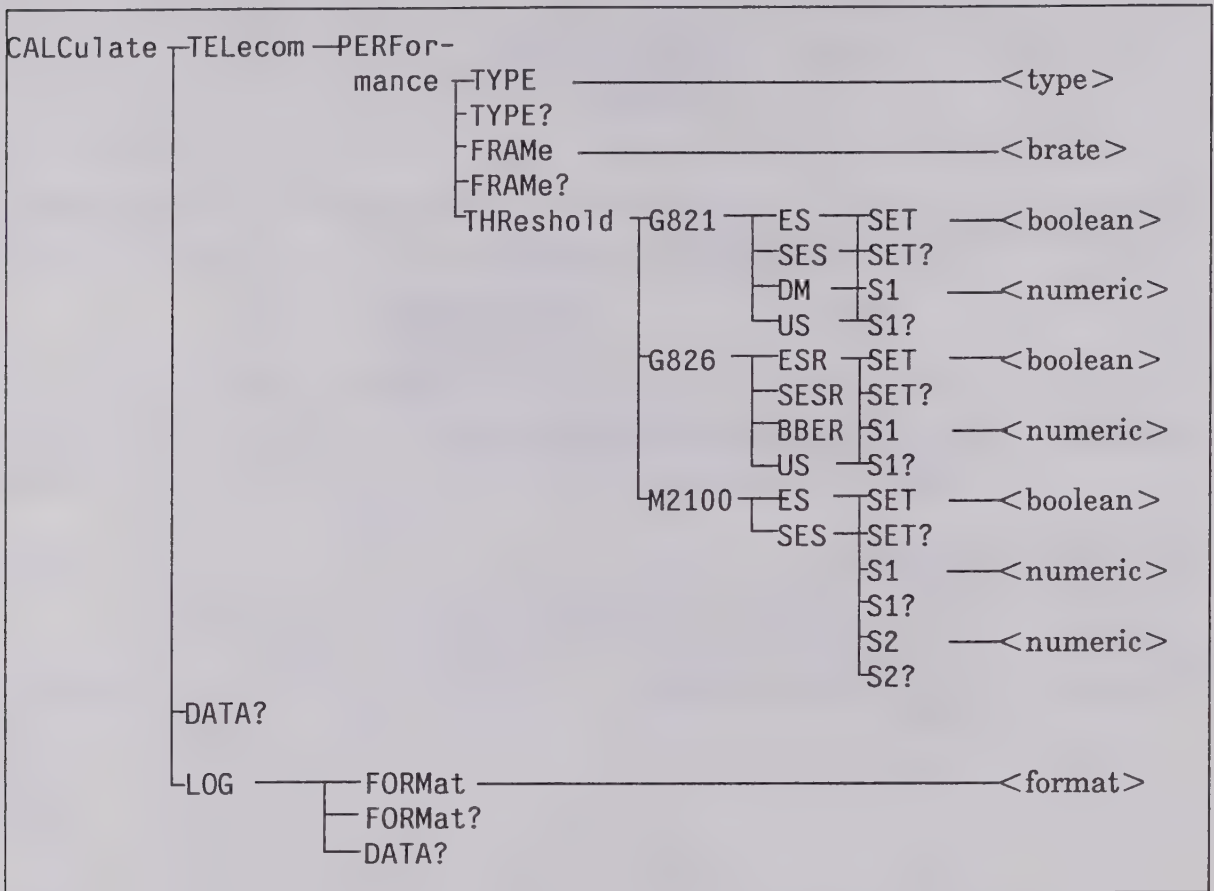
■ :ROUTe:INPut:SElect?

- Response <port>= <CHARACTER RESPONSE DATA>
- Function Queries the signal input connector when the receiver signal is 2M.
- Example > :ROUTe:INPut:SElect?


```
< PORT1
```

9.3.6 CALCulate subsystem

The CALCulate subsystem is used to set the performance measurement and to display the measured results.



■ :CALCulate:TELEcom:PERFORMANCE:TYPE <type>

- Parameter <type> = <CHARACTER PROGRAM DATA>

OFF	Does not measure performance.
G821	G.821 performance measurement
G826	G.826 performance measurement
M2100	M.2100 performance measurement
- Function Sets one type of performance measurement.
- Example Set the performance measurement to M.2100.
 > :CALCulate:TELEcom:PERFORMANCE:TYPE M2100

■ :CALCulate:TELecom:PERFormance:TYPE?

- Response <type>= <CHARACTER RESPONSE DATA>
- Function Queries the type which has been set for the performance measurement
- Example > :CALCulate:TELecom:PERFormance:TYPE?
 <M2100

■ :CALCulate:TELecom:PERFormance:FRAME <brate>

- Parameter <brate>= <CHARACTER PROGRAM DATA>

M139	139 Mb/s
M34	34 Mb/s
M8	8 Mb/s
M2	2 Mb/s
- Function Sets the frame for executing the performance measurement
 When the Demultiplexer is set to OFF, the bit rate frame being used is selected irrespective of this setting.
- Limitation Not valid in the following cases:
 - ▶ The <OSERvice> parameter is set in the :SENSe:TELecom:MMODE command.
 - ▶ The <OFF> parameter is set in the CALCulate:TELecom:PERFormance:TYPE command.
 - ▶ The MUX/DEMUX option is not incorporated, or the DEMUX (Demultiplexer) is set to OFF
- Example Executes performance measurement at 34 Mb/s.
 > :CALCulate:TELecom:PERFormance:FRAME M34

■ :CALCulate:TELecom:PERFormance:FRAME?

- Response <brate>= <CHARACTER RESPONSE DATA>
- Function Queries the frame which has been selected for performance measurement. When queried, the set value is returned irrespective of the bit rate being used.
- Example > :CALCulate:TELecom:PERFormance:FRAME?
 <M34

■ :CALCulate:TELecom:PERFormance:THReshold:G821:ES:SET <boolean>**■ :CALCulate:TELecom:PERFormance:THReshold:G821:SES:SET <boolean>****■ :CALCulate:TELecom:PERFormance:THReshold:G821:DM:SET <boolean>****■ :CALCulate:TELecom:PERFormance:THReshold:G821:US:SET <boolean>**

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>

OFF or 0	Does not make a decision
ON or 1	Makes a decision
- Function Switches the G.827 threshold-value decision for performance measurement to ON or OFF.
 For commands listed above, switches the decisions relating to %ES, %SES, %DM and %US to ON or OFF.

- Limitation Not valid unless parameter <G821> is set in the :CALCulate:TELEcom:PERFormance:TYPE command.

- Example Set the G.821 threshold value %ES decision to ON
>:CALCulate:TELEcom:PERFormance:THReshold:G821:ES:SET ON

■ :CALCulate:TELEcom:PERFormance:THReshold:G821:ES:SET?
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:SES:SET?
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:DM:SET?
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:US:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
0 or 1

- Function Queries whether the G821 threshold-value decision for performance measurement is set or not.

- Example >:CALCulate:TELEcom:PERFormance:THReshold:G821:ES:SET?
< 1

■ :CALCulate:TELEcom:PERFormance:THReshold:G821:ES:S1 <numeric>
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:SES:S1 <numeric>
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:DM:S1 <numeric>
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:US:S1 <numeric>

- Parameter <numeric>=<NR2 NUMERIC PROGRAM DATA>
0.0000 to 99.9999 Step value: 0.0001

- Function Sets the threshold value when the G.821 performance is determined.
For the commands listed above, sets the threshold values of %ES, %SES, %DM, and %US.
A parameter numeric value is expressed in real number format.

- Limitation Not valid if the <OFF> parameter is set in the :CALCulate:TELEcom:PERFormance:THReshold:G821:...:SET command

- Example Set the G.821 %ES threshold value to 50.0000.
>:CALCulate:TELcom:PERFormance:THReshold:G821:ES:S1 50.0000

■ :CALCulate:TELEcom:PERFormance:THReshold:G821:ES:S1?
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:SES:S1?
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:DM:S1?
 ■ :CALCulate:TELEcom:PERFormance:THReshold:G821:US:S1?

- Response <numeric>=<NR2 NUMERIC RESPONSE DATA>
- Function Queries the threshold value which has been set for G.821 performance decision.
- Example >:CALCulate:TELcom:PERFormance:THReshold:G821:ES:S1?
< 50.0000

- :CALCulate:TELEcom:PERFormance:THReshold:G826:ESR:SET? <boolean>
- :CALCulate:TELEcom:PERFormance:THReshold:G826:SESR:SET? <boolean>
- :CALCulate:TELEcom:PERFormance:THReshold:G826:BBER:SET? <boolean>
- :CALCulate:TELEcom:PERFormance:THReshold:G826:US:SET? <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>

OFF or 0	Does not make a decision
ON or 1	Makes a decision
- Function Switches the G.826 threshold-value decision for performance measurement to ON or OFF.
 For the commands listed above, switches the decisions relating to ESR, SESR, BBER, and %US to ON or OFF.
- Limitation Not valid unless parameter <G826> is set in the :CALCulate:TELEcom:PERFormance:TYPE command.
- Example Set the G.826 threshold value ESR decision to ON.
 >:CALCulate:TELEcom:PERFormance:THReshold:G826:ESR:SET ON

- :CALCulate:TELEcom:PERFormance:THReshold:G826:ESR:SET?
- :CALCulate:TELEcom:PERFormance:THReshold:G826:SESR:SET?
- :CALCulate:TELEcom:PERFormance:THReshold:G826:BBER:SET?
- :CALCulate:TELEcom:PERFormance:THReshold:G826:US:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether the decision for G.826 performance threshold value is set or not.
- Example >:CALCulate:TELEcom:PERFormance:THReshold:G826:ESR:SET?
 < 1

- :CALCulate:TELEcom:PERFormance:THReshold:G826:ESR:S1 <numeric>
- :CALCulate:TELEcom:PERFormance:THReshold:G826:SESR:S1 <numeric>
- :CALCulate:TELEcom:PERFormance:THReshold:G826:BBER:S1 <numeric>
- :CALCulate:TELEcom:PERFormance:THReshold:G826:US:S1 <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1.00E-09 to 9.99E-01	Setting range for ESR, SESR, and BBER
Where, mantissa part 1.00 to 9.99	Step value: 0.01
Exponent part -09 to -01	Step value: 1
0.0000 to 99.9999	%US setting range Step value: 0.0001
- Function Function Sets the threshold value when the G.826 performance is determined.
 For the commands listed above, sets the threshold values of ESR, SESR, BBER, and %US.
 A parameter numeric value is expressed in real number format.
- Limitation Not valid if the <OFF> parameter is set in the :CALCulate:TELEcom:PERFormance:THReshold:G826:...:SET command

- Example Set the G.826 ESR threshold value to 5.00E-05.
 >:CALCulate:TELEcom:PERFormance:THReshoId:G826:ESR:S1
 5.00E-05

■ :CALCulate:TELEcom:PERFormance:THReshoId:G826:ESR:S1?
■ :CALCulate:TELEcom:PERFormance:THReshoId:G826:SESR:S1?
■ :CALCulate:TELEcom:PERFormance:THReshoId:G826:BBER:S1?
■ :CALCulate:TELEcom:PERFormance:THReshoId:G826:US:S1?

- Response <numeric>= <NR3 NUMRIC RESPONSE DATA>
- Function Queries the threshold value which has been set for G.826 performance decision..
- Example >:CALCulate:TELEcom:PERFormance:THReshoId:G826:ESR:S1?
 < 5.00E-05

■ :CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:SET <boolean>
■ :CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:SET <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Does not make a decision
 ON or 1 Makes a decision
- Function Function Switches the M.2100 threshold-value decision for performance measurement to ON or OFF.
 For the commands listed above, switches the decisions relating to ES and SES to ON or OFF.
- Limitation Not valid unless parameter <M2100> is set in the :CALCulate:TELEcom:PERFormance:TYPE command.
- Example Set the M.2100 threshold value ES decision to ON
 >:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:SET ON

■ :CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:SET?
■ :CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether the decision for M.2100 performance threshold value is set or not.
- Example >:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:SET?
 < 1

:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:S1 <S1>

:CALCulate:TELEcom:PERFormance:THReshold:M2100:SES:S1 <S1>

```
■ :CALCulate:TELEcom:PERFormance:THReshold:M2100:ES:S2 <S2>
```

```
■ :CALCulate:TELEcom:PERFormance:THReshold:M2100:SES:S2 <S2>
```

- Parameter <S1>=<DECIMAL NUMERIC PROGRAM DATA>

0 to 99,998 Step value: 1

<S2>= <DECIMAL NUMERIC PROGRAM DATA>

1 to 99,999 Step value: 1

Set this with $\langle S1 \rangle \leftarrow \langle S2 \rangle$.

- | | |
|--------------|---|
| ● Function | <p>Sets the threshold value when the M.2100 performance is determined.</p> <p>For the commands listed above, sets the S1 (threshold lower limit value) and S2 (threshold upper limit value) of ES and SES.</p> <p>If the parameter is set as $\langle S1 \rangle \geq \langle S2 \rangle$, then the $\langle S2 \rangle$ value is set to $\langle S1 \rangle + 1$.</p> |
| ● Limitation | Not valid if the $\langle OFF \rangle$ parameter is set in the :CALCulate:TELEcom:PERFormance:THReshold:M2100:...:SET command. |
| ● Example | <p>Set the M.2100 ES threshold value so that the upper and lower limit are 10 and 4 respectively.</p> <pre>> :CALCulate:TELEcom:PERFormance:THReshold:M2100:ES:S1 4 > :CALCulate:TELEcom:PERFormance:THReshold:M2100:ES:S2 10</pre> |

:CALCulate:TELEcom:PERFormance:THReshold:M2100:ES:S1?

:CALCulate:TELEcom:PERFormance:THReshold:M2100:SES:S1?

:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:S2?

:CAICulate:TELeom:PERFormance:THReshold:M2100:SES:S2?

- Response <S1>, <S2> = <NR1 NUMERIC RESPONSE DATA>
- Function Queries the M.2100 performance threshold value.
- Example > :CALCulate:TELEcom:PERFORMANCE:THReshoId:M2100:ES:S1?<4

■ :CALCulate:DATA? <string>

- Parameter <string>= <STRING PROGRAM DATA>

```
...["CURRENT:"]<result1>"
```

Current measured result
(Upon error measurement or Jitter Manual
measurement)

```
"LAST:<result1>"
```

Last measured result
(Upon error measurement or Jitter Manual
measurement)

"result2"

Sweep mask, Transfer and Jitt/Freq
measured result

Refer to Table 9-4 for the contents of `<result1>` and Table 9-5 for the contents of `<result2>`.

- Response <string>=<STRING RESPONSE DATA>

Refer to Tables 9-4 and 9-5 (refer to 9.1 response format for Forms 1 to 11).

- **Function** Queries the measured results corresponding to a parameter.

- Example Query the bit error EC current value.

```
>:CALCulate:DATA? "CURRENT:EC:BIT"
or :CALCulate:DATA? "EC:BIT"    (CURRENT can be abbreviated.)
< "      892"
```

Queries the decision (last result) of the M.2100 performance measurement.

```
>:CALCulate:DATA?"LAST:M2100:TEST"
< "    Degraded"
```

Table 9-4 Query Contents of Measured Result (1/2)

Item		<result>	Response format
EC	Bit error Code error Frame error	"EC:BIT" "EC:CODE" "EC:FRAME:M139" "EC:FRAME:M34" "EC:FRAME:M8" "EC:FRAME:M2"	Form1
	CRC-4	"EC:CRC"	
ER	Bit error Code error Frame error	"ER:BIT" "ER:CODE" "ER:FRAME:M139" "ER:FRAME:M34" "ER:FRAME:M8" "ER:FRAME:M2"	Form2
	CRC-4	"ER:CRC"	
ES	Bit error Code error Frame error	"ES:BIT" "ES:CODE" "ES:FRAME:M139" "ES:FRAME:M34" "ES:FRAME:M8" "ES:FRAME:M2"	Form1
	CRC-4	"ES:CRC"	
G.821	%ES %SES %DM %US Test	"G821:ES" "G821:SES" "G821:DM" "G821:US" "G821:TEST"	Form3 Form5
G.826	ESR	"G826:ESR"	Form2
	SESR	"G826:SESR"	Form2
	BBER	"G826:BBER"	Form2
	%US	"G826:US"	Form3
	Test	"G826:TEST"	Form5

For Form 1 to Form 5, see paragraph 9.1 Response Format.

Table 9-4 Query Contents of Measured Result (2/2)

Item		<result>	Response format
M.2100	ES	"M2100:ES"	Form1
	SES	"M2100:SES"	
	Test	"M2100:TEST"	Form5
Power Fail		"ASEConds:POWer"	Form1
LOS		"ASEConds:LOS"	Form1
AIS	139M	"ASEConds:AIS:M139"	Form1
	34M	"ASEConds:AIS:M34"	
	8M	"ASEConds:AIS:M8"	
	2M	"ASEConds:AIS:M2"	
LOF	139M	"ASEConds:LOF:M139"	Form1
	34M	"ASEConds:LOF:M34"	
	8M	"ASEConds:LOF:M8"	
	2M	"ASEConds:LOF:M2"	
RA	139M	"ASEConds:RA:M139"	Form1
	34M	"ASEConds:RA:M34"	
	8M	"ASEConds:RA:M8"	
	2M	"ASEConds:RA:M2"	
Pattern Sync Loss		"ASEConds:PATtern"	Form1
LOF MF		"ASEConds:LOF:MF"	Form1
RA MF		"ASEConds:RA:MF"	Form1
Frequency Offset		"FREQuency:OFFSet"	Form4
Rx amplitude		"RX:AMPLitude"	Form7
Rx + peak		"RX:PPEak"	Form7
Rx – peak		"RX:MPEak"	Form7
Hit count		"HIT:COUNt"	Form1
Hit seconds		"HIT:SEConds"	Form1
%Hit FS		"HIT:FS"	Form3
Status		"MANual:STATus"	Form10

Table 9-5 Query Contents of Measured Result <result2>

Item	<result2>	Response format
Measured result of each point in Sweep mask measurement	"SMASk:POINt1", "SMASk:POINt2", ..., "SMASk:POINt10" • Queries the measured results (Error/No Error) of points 1 through 10 in the table.	Form11
Measured result of each point in Transfer measurement (when Dspl:Cal (Table))	"CTABle:POINt1", "CTABle:POINt2", ..., "CTABle:POINt10" • Queries the values of points 1 through 10 displayed in the Transfer screen when Dspl:Cal (Table). • Returns the calibration value of the bit rate concurrently selected.	Form8
Measured result of each point in Transfer measurement (when Dspl:Meas (Table))	"MTABle:POINt1", "MTABle:POINt2", ..., "MTABle:POINt10" • Queries the values of points 1 through 10 displayed in the Transfer screen when Dspl:Meas (Table).	Form8
Transfer measurement status	"TRANSfer:STATUs"	Form10
Max jitter offset in Jitt/Freq measurement	"MAX:OFFSet"	Form9
Max jitter magnitude in Jitt/Freq measurement	"MAX:AMPLitude"	Form7
Measured result of each point in Jiit/Freq measurement	"JFRequency:POINt1", ..., "JFRequency:POINt101" • Queries the graph plot values in Jitt/Freq measurement. (Designates the measurement starting point as <POINt1> and queries up to the maximum <POINt101>).	Form7
Jitt/Freq measurement status	"JFRequency:STATUs"	Form10

■ :CALCulate:LOG:FORMat <format>

- Parameter <format>= <CHARACTER PROGRAM DATA>

EC	Error count
ER	Error rate
- Function Sets the log data display item which indicates the response display item after the :CALCulate:LOG:DATA? query command has been executed.
- Example Set the log data display item to ER.


```
>:CALCulate:LOG:FORMat ER
```

■ :CALCulate:LOG:FORMat?

- Response <format>= <CHARACTER RESPONSE DATA>
- Function Queries the log data display item
- Example >:CALCulate:LOG:FORMat?


```
< ER
```

■ :CALCulate:LOG:DATA? <start>,<stop>

- Parameter <start>= <DECIMAL NUMERIC PROGRAM DATA>

1 to final line of log data	First line number for display
-----------------------------	-------------------------------

 <stop>= <DECIMAL NUMERIC PROGRAM DATA>

1 to final line of log data	Last line number for display
-----------------------------	------------------------------

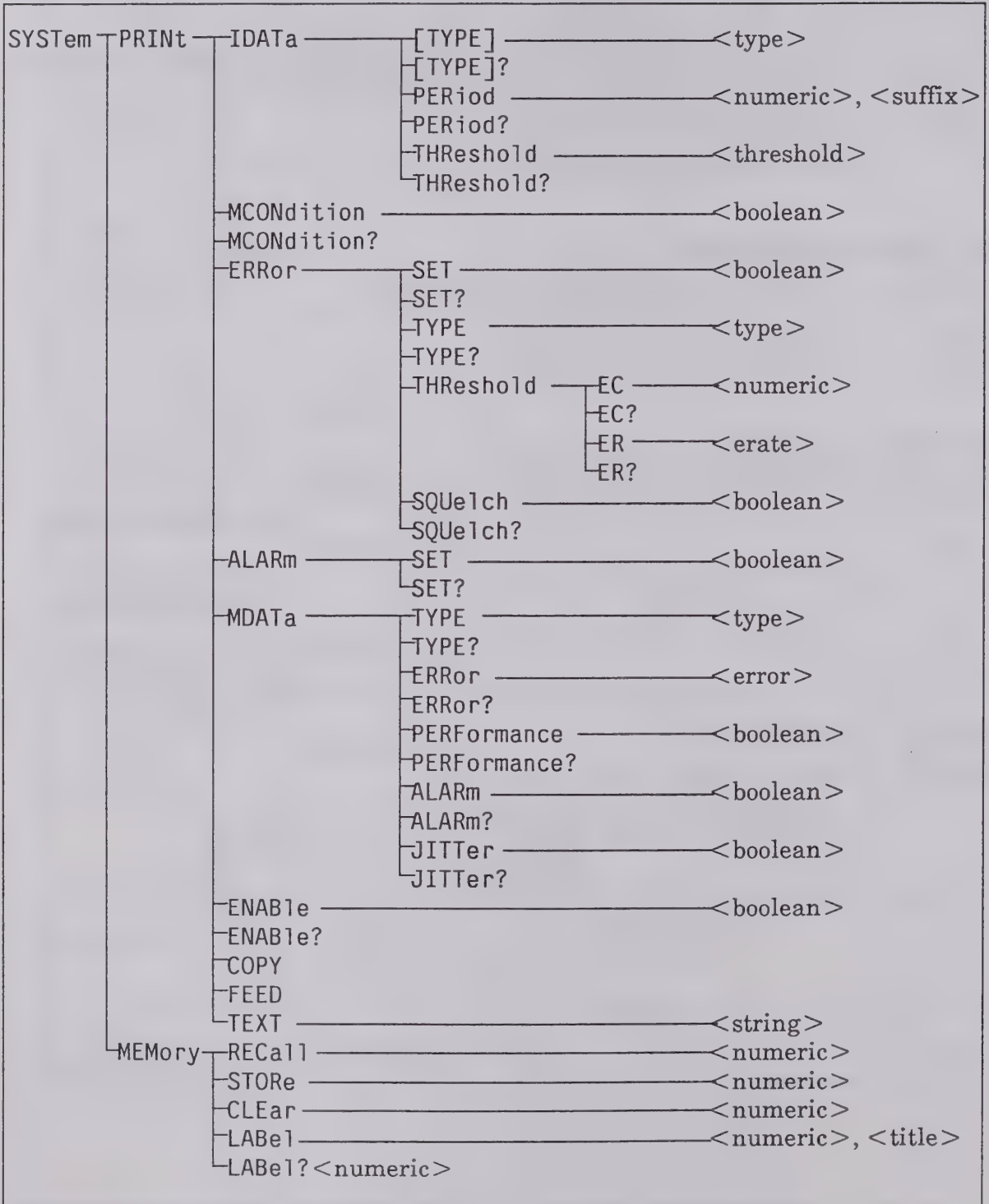
 <string>= <STRING RESPONSE DATA>

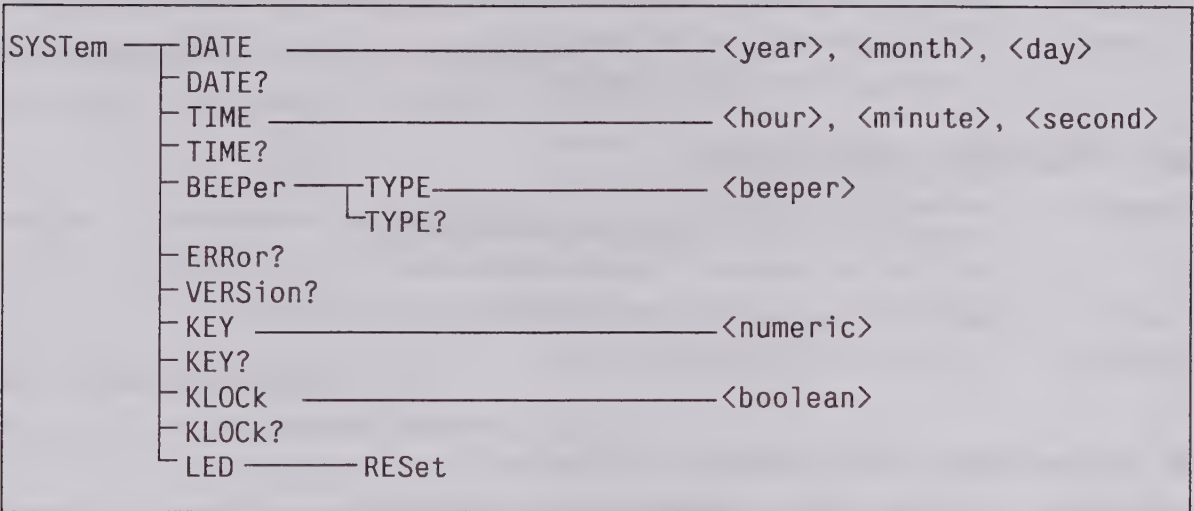
Form6 (See response format described previously)	
Data ranging between <start> line and <stop> line in the log data	
- Function Displays log data.
- Example Query the log data from line 1 through line 3.


```
>:CALCulate:LOG:DATA? 1,3
<"10-21 12:34:56 Bit Err          1.08E+13 ",
  "                               Code Err    1.09E+08 ",
  "                               FRM 34M Err    122 "
```

9.3.7 SYSTem subsystem

The SYSTem subsystem is sued to set printer, memory, buzzer etc.





■ :SYSTem:PRINt:IDATa:[TYPE] <type>

- Parameter <type>= <CHARACTER PROGRAM DATA>

OFF	Does not output intermediate data.
INDividual	Value from just prior to intermediate data
ACCumulative	Accumulated value from measurement start
- Function Selects the value type printed from intermediate data.
- Example Print out the value from just prior to intermediate data.


```
>:SYSTem:PRINt:IDATa:TYPE INDividual
or :SYSTem:PRINt:IDATa INDividual
```

■ :SYSTem:PRINt:IDATa[:TYPE]?

- Response <type>= <CHARACTER RESPONSE DATA>
- Function Queries the value type printed from intermediate data.
- Example

```
>:SYSTem:PRINt:IDATa:TYPE?
or :SYSTem:PRINt:IDATa?
< IND
```

■ :SYSTem:PRINt:IDATa:PERiod <numeric>,<suffix>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1 to 99	Step value: 1
---------	---------------
- <suffix>= <CHARACTER PROGRAM DATA>

d	day
h	hour
m	minute
s	second

- **Function** Sets the print period for intermediate data.
- **Limitation** Not valid if the <OFF> parameter is set in the :SYSTEM:PRINT:IDATA[:TYPE].
- **Example** Set the intermediate data print period to 1 hour.
> :SYSTEM:PRINT:IDATA:PERiod 1,h

■ :SYSTEM:PRINT:IDATA:PERiod?

- **Response** <numeric>, <suffix>
<numeric>= <NR1 NUMERIC RESPONSE DATA>
<suffix>= <CHARACTER RESPONSE DATA>
- **Function** Queries the print period for intermediate data.
- **Example** > :SYSTEM:PRINT:IDATA:PERiod?
<1,h

■ :SYSTEM:PRINT:IDATA:THReshold <threshold>

- **Parameter** <threshold>= <CHARACTER PROGRAM DATA>

OFF	Always prints out intermediate data.
EALarm	Prints out intermediate data when an error has occurred or when there is a change in alarm.
- **Function** Sets the threshold for printing intermediate data.
- **Example** Always print out intermediate data.
> :SYSTEM:PRINT:IDATA:THReshold OFF

■ :SYSTEM:PRINT:IDATA:THReshold?

- **Response** <threshold>= <CHARACTER RESPONSE DATA>
- **Function** Queries the threshold for printing intermediate data.
- **Example** > :SYSTEM:PRINT:IDATA:THReshold?
<OFF

■ :SYSTEM:PRINT:MCONdition <boolean>

- **Parameter** <boolean>= <BOOLEAN PROGRAM DATA>

OFF or 0	Does not print out measurement conditions.
ON or 1	Prints our measurement conditions.
- **Function** Switches the function for printing measurement conditions to ON or OFF.
- **Example** Set the function for printing measurement conditions to ON.
> :SYSTEM:PRINT:MCONdition ON

■ :SYSTEM:PRINT:MCONdition?

- **Response** <NR1 NUMERIC RESPONSE DATA>
0 or 1
- **Function** Queries whether the function for printing measurement conditions is set or not.

● Example >:SYSTem:PRINt:MCONdition?
 < 1

■ :SYSTem:PRINt:ERROr:SET <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Does not print out to indicate an error occurrence.
 ON or 1 Prints out to indicate an error occurrence.
- Function Switches the function for printing error occurrence to ON or OFF.
- Example Print out to indicate an error occurrence.
 >:SYSTem:PRINt:ERROr:SET ON

■ :SYSTem:PRINt:ERROr:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether the function for printing an error occurrence is set or not.
- Example >:SYSTem:PRINt:ERROr:SET?
 < 1

■ :SYSTem:PRINt:ERROr:TYPE <type>

- Parameter <type>= <CHARACTER PROGRAM DATA>
 EC Error count
 ER Error rate
- Function Sets the format for printing an error occurrence.
- Limitation Not valid if the <OFF> parameter is set in the :SYSTem:PRINt:ERROr:SET command.
- Example Print out in the error count format.
 >:SYSTem:PRINt:ERROr:TYPE EC

■ :SYSTem:PRINt:ERROr:TYPE?

- Response <type>= <CHARACTER RESPONSE DATA>
- Function Queries the format which has been set for printing an error occurrence.
- Example >:SYSTem:PRINt:ERROr:TYPE?
 < EC

■ :SYSTem:PRINt:ERROr:THReshold:EC <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 0 OFF
 1 > 1
 10 > 10
 100 > 100
 1000 >1000
- Function Sets the threshold value of the error count printed when there has been an error occurrence.

Printing is executed when the number of errors which occur in one second exceeds the set number.

- **Limitation** Not valid in the following cases:
 - ▶ The <OFF> parameter is set in the :SYSTEM:PRINT:ERROR:SET command
 - ▶ The <ER> parameter is set in the :SYSTEM:PRINT:ERROR:TYPE command.
- **Example** Print out when errors exceeding 100 counts per second have occurred.


```
> :SYSTEM:PRINT:ERROR:THRESHOLD:EC 100
```

■ :SYSTEM:PRINT:ERROR:THRESHOLD:EC?

- **Response** <numeric>= <CHARACTER RESPONSE DATA>
- **Function** Queries the threshold value of the error count printed when there has been an error occurrence.
- **Example**

```
> :SYSTEM:PRINT:ERROR:THRESHOLD:EC?
< 100
```

■ :SYSTEM:PRINT:ERROR:THRESHOLD:ER <erate>

- **Parameter** <erate>= <CHARACTER PROGRAM DATA>

OFF	
E_3	>1E-3
E_4	>1E-4
E_5	>1E-5
E_6	>1E-6
- **Function** Sets the threshold value of the error rate printed when there has been an error occurrence.

Printing is executed when the error rate per second exceeds the set value.
- **Limitation** Not valid in the following cases:
 - ▶ The <OFF> parameter is set in the :SYSTEM:PRINT:ERROR:SET command
 - ▶ The <EC> parameter is set in the :SYSTEM:PRINT:ERROR:TYPE command
- **Example** Print out when errors exceeding 1E-4 counts per second have occurred.


```
> :SYSTEM:PRINT:ERROR:THRESHOLD:ER E_4
```

■ :SYSTEM:PRINT:ERROR:THRESHOLD:ER?

- **Response** <erate>= <CHARACTER RESPONSE DATA>
- **Function** Queries the threshold value of the error rate printed when there has been an error occurrence.
- **Example**

```
> :SYSTEM:PRINT:ERROR:THRESHOLD:ER?
< E_4
```

■ :SYSTEM:PRINT:ERROR:SQUELCH <boolean>

- **Parameter** <boolean>= <BOOLEAN PROGRAM DATA>

OFF or 0	Continues printing.
ON or 1	Stops printing.

- Function Switches the function to stop printing at continuous error occurrence to ON or OFF.
- Limitation Not valid if the <OFF> parameter is set in the :SYSTem:PRINt:ERRor:SET command.
- Example Stop printing when a continuous error occurs.
> :SYSTem:PRINt:ERRor:SQUelch ON

■ :SYSTem:PRINt:ERRor:SQUelch?

- Response <NR1 NUMERIC RESPONSE DATA>
0 or 1
- Function Queries whether printing is stopped or not when a continuous error occurs.
- Example > :SYSTem:PRINt:ERRor:SQUelch?
< 1

■ :SYSTem:PRINt:ALARm:SET <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
OFF or 0 Does not print out alarm data.
ON or 1 Prints out at alarm occurrence/recovery.
- Function Switches the function for printing at alarm occurrence to ON or OFF.
- Example Print out at alarm occurrence/recovery.
> :SYSTem:PRINt:ALARm:SET ON

■ :SYSTem:PRINt:ALARm:SET?

- Response <NR1 NUMERIC RESPONSE DATA>
0 or 1
- Function Queries whether the function for printing at alarm occurrence is set or not.
- Example > :SYSTem:PRINt:ALARm:SET?
< 1

■ :SYSTem:PRINt:MDATa:TYPE <type>

- Parameter <type>= <CHARACTER PROGRAM DATA>
ALL Prints out all the measured result data
NONE Does not print out the measured result data.
SELEct Selects and prints out the measured result data.
- Function Selects the content for printing the measured result data.
- Example Select and print out the measured result data.
> :SYSTem:PRINt:MDATa:TYPE SELEct

■ :SYSTem:PRINt:MDATa:TYPE?

- Response <type>= <CHARACTER RESPONSE DATA>
- Function Queries the content for printing the measured result data.

- Example `> :SYSTem:PRINT:MDATa:TYPE?`
 `< SEL`

■ `:SYSTem:PRINT:MDATa:ERRor <error>`

- Parameter `<error>= <STRING PROGRAM DATA>`
 "NONE" Does not display error data.
 "ALL" All error data
 "CODE" Code error data
 "BIT" Bit error data
 "CRC" CRC error data
 "FRAME:M139" 139M frame error
 "FRAME:M34" 34M frame error
 "FRAME:M8" 8M frame error
 "FRAME:M2" 2M frame error
- Function Selects the type of error for printing
- Limitation Not valid unless the parameter `<SElect>` is set in the `:SYSTem:PRINT:MDATa:TYPE` command
- Example Print out the error data for the 139M frame error.
 `> :SYSTem:PRINT:MDATa:ERRor "FRAME:M139"`

■ `:SYSTem:PRINT:MDATa:ERRor?`

- Response `<error>= <CHARACTER RESPONSE DATA>`
- Function Queries the type of error which has been selected for printing.
- Example `> :SYSTem:PRINT:MDATa:ERRor?`
 `< "FRAME:M139"`

■ `:SYSTem:PRINT:MDATa:PERFformance <boolean>`

- Parameter `<boolean>= <BOOLEAN PROGRAM DATA>`
 OFF or 0 Does not print out the measured performance results.
 ON or 1 Print out the measured performance results.
- Function Switches the function for printing the measured performance results to ON or OFF.
- Limitation Not valid unless the parameter `<SElect>` is set in the `:SYSTem:PRINT:MDATa:TYPE` command.
- Example Print out the measured performance results.
 `> :SYSTem:PRINT:MDATa:PERFformance ON`

■ `:SYSTem:PRINT:DMATa:PERFformance?`

- Response `<NR1 NUMERIC RESPONSE DATA>`
 0 or 1
- Function Queries whether the function for printing the measured performance results is set or not.
- Example `> :SYSTem:PRINT:DMATa:PERFformance?`
 `< 1`

■ :SYSTem:PRINT:MDATa:ALARm <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Does not print out the measured alarm results
 ON or 1 Prints out the measured alarm results.
- Function Switches the function for printing the measured alarm results to ON or OFF.
- Limitation Not valid unless the parameter <SElect> is set in the :SYSTem:PRINT:MDATa:TYPE command.
- Example Print out the measured alarm results.
 > :SYSTem:PRINT:MDATa:ALARm ON

■ :SYSTem:PRINT:MDATa:ALARm?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether the function for printing the measured alarm results is set or not.
- Example > :SYSTem:PRINT:MDATa:ALARm?
 < 1

■ :SYSTem:PRINT:MDATa:JITTer <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Do not print the jitter measured results.
 ON or 1 Print the jitter measured results.
- Function Sets whether or not to print the jitter measured results.
- Limitation Not valid unless <SElect> is specified in :SYSTem:PRINT:MDATa:TYPE command.
 Not valid unless the jitter option is incorporated.
- Example Print out the measured jitter results.
 > :SYSTem:PRINT:MDATa:JITTer ON

■ :SYSTem:PRINT:MDATa:JITTer?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether the function for printing the measured jitter results is set or not.
- Limitation Not valid unless the jitter option is incorporated.
- Example > :SYSTem:PRINT:MDATa:JITTer?
 < 1

■ :SYSTem:PRINT:ENABLE <boolean>

- Parameter <boolean>= <BOOLEAN PROGRAM DATA>
 OFF or 0 Printer OFF
 ON or 1 Printer ON
- Function Switches the printer to ON or OFF. (Printer key)
- Example Enable the printer to print out.
 > :SYSTem:PRINT:ENABLE ON

■ :SYSTem:PRINT:ENABLE?

- Response <NR1 NUMERIC RESPONSE DATA>
 0 or 1
- Function Queries whether the printer is set to ON or not.
- Example > :SYSTem:PRINT:ENABLE?
 < 1

■ :SYSTem:PRINT:COPY

- Parameter None
- Function Copies the contents currently displayed on the screen. (Copy key)
- Example Copy the screen contents.
 > :SYSTem:PRINT:COPY

■ :SYSTem:PRINT:FEED

- Parameter None
- Function Feeds the printer paper.
 Feeds the paper line by line when one command is executed.
- Example Feed the printer paper.
 > :SYSTem:PRINT:FEED

■ :SYSTem:PRINT:TEXT <string>

- Parameter <string>= <STRING PROGRAM DATA>
 "Each character string" (A maximum of 40 characters including "")
- Function Outputs the parameter-specified string to the built-in printer.
- Example Output "--- 139M TEST ---" to the built-in printer.
 > :SYSTem:PRINT:TEXT "--- 139M TEST ---"

■ :SYSTem:MEMory:RECall <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 0 to 20
 0: Initial (factory setting at shipment)
- Function Recalls data from memory.
 This command is equivalent to the *RCL common command.
- Limitation Not valid if a memory number in which no data is written is specified.
- Example Recall data from memory location number 1.
 > :SYSTem:MEMory:RECall 1

■ :SYSTem:MEMory:STORe <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 20
- Function Writes data in memory.
 This command is equivalent to the *SAV common command.
- Example Write data in memory location number 3.
 > :SYSTem:MEMory:STORe 3

■ :SYSTem:MEMory:CLEAr <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 20
- Function Clears the data stored in the memory.
- Limitation Not valid if a memory number in which no data is written is specified.
- Example Clear the data stored in memory location number 3.
 > :SYSTem:MEMory:CLEAr 3

■ :SYSTem:MEMory:LABel <numeric>,<title>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 20
 <title>= <STRING PROGRAM DATA>
 Memory name
- Function Names memory.
- Limitation Not valid if a memory number in which no data is written is specified.
- Example Name memory number 1 as "139M TEST".
 > :SYSTem:MEMory:LABel 1, "139M TEST"

■ :SYSTem:MEMory:LABel? <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 to 20
- Response <title>= <STRING RESPONSE DATA>
- Function Queries memory's name.
- Example Query the name of memory number 1.
 > :SYSTem:MEMory:LABel? 1
 > "139M TEST"

■ :SYSTem:DATE <year>,<month>,<day>

- Parameter <DECIMAL NUMERIC PROGRAM DATA>
 <year> = 1993 to 2092
 <month> = 1 to 12
 <day> = 1 to 31
- Function Sets the current date.
- Example Set the current date (July 28, 1993).
 > :SYSTem:DATE 1993,7,28

■ :SYSTem:DATE?

- Response <year>,<month>,<day>
 = <NR1 NUMERIC RESPONSE DATA>
- Function Queries the current date.
- Example > :SYSTem:DATE?
 < 1993,7,28

■ :SYSTem:TIME <hour>,<minute>,<second>

- Parameter <DECIMAL NUMERIC PROGRAM DATA>
 <hour> = 0 to 23
 <month> = 0 to 59
 <second> = 0 to 59
- Function Sets the current time.
- Example Set the current time (14:00:00)
 > :SYSTem:TIME 14,0,0

■ :SYSTem:TIME?

- Response <hour>,<minute>,<second>
 = <NR1 NUMERIC RESPONSE DATA>
- Function Queries the current time
- Example > :SYSTem:TIME?
 < 14,0,0

■ :SYSTem:BEEPer:TYPE <beeper>

- Parameter <beeper>= <CHARACTER PROGRAM DATA>

OFF	Does not sound a buzzer.
ERRor	An error has occurred
ALARm	An alarm has occurred
EALARm	Both an error and an alarm have occurred
- Function Sets one of the conditions for sounding a buzzer.
- Example Sound a buzzer when an alarm has occurred.
 > :SYSTem:BEEPer:TYPE ALARm

■ :SYSTem:BEEPer:TYPE?

- Response <beeper>= <CHARACTER RESPONSE DATA>
- Function Queries which condition has been set for sounding a buzzer.
- Example > :SYSTem:BEEPer:TYPE?
 <ALAR

■ :SYSTem:ERRor?

- Response <error/event_number>,<error/event_description>
 <error/event_number>= <NR1 NUMERIC RESPONSE DATA>
 – 32768 to 32767
 <error/event_description>= <STRING RESPONSE DATA>
- Function Queries error status.
 (For error message details, see Appendix C.)
- Example > :SYSTem:ERRor?
 <–102,"Syntax error"

■ :SYSTem:VERSion?

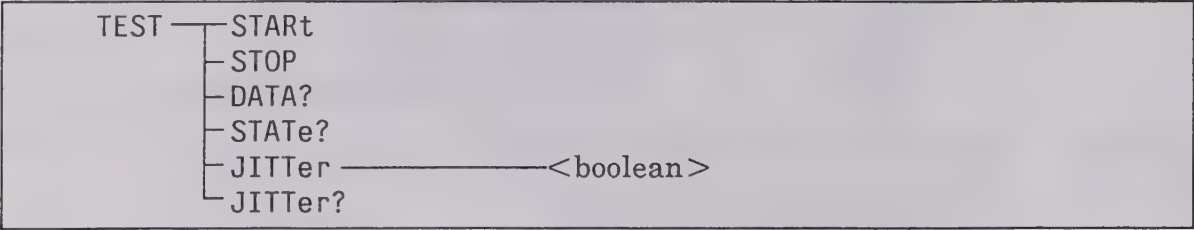
- Response <version>
 <version>= <NR2 NUMERIC RESPONSE DATA>
 YYYY.V YYYY:year
 V :revision number
- Function Queries the version number of this instrument conforming to SCPI.
- Example > :SYSTem:VERSion?
 <1993.0

■ :SYSTem:LED:RESet

- Parameter None
- Function Resets the LED history.
- Example > SYSTem:LED:RESet

9.3.8 TEST subsystem

The TEST subsystem is used to execute and display the results of self-test.



■ :TEST:START

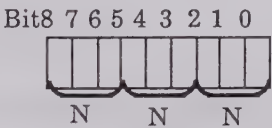
- Parameter None
- Function Starts executing self-test.
- Limitation Not valid if executed during measurement.
- Example > :TEST:START

■ :TEST:STOP

- Parameter None
- Function Stops executing self-test.
- Example > :TEST:STOP

■ :TEST:DATA?

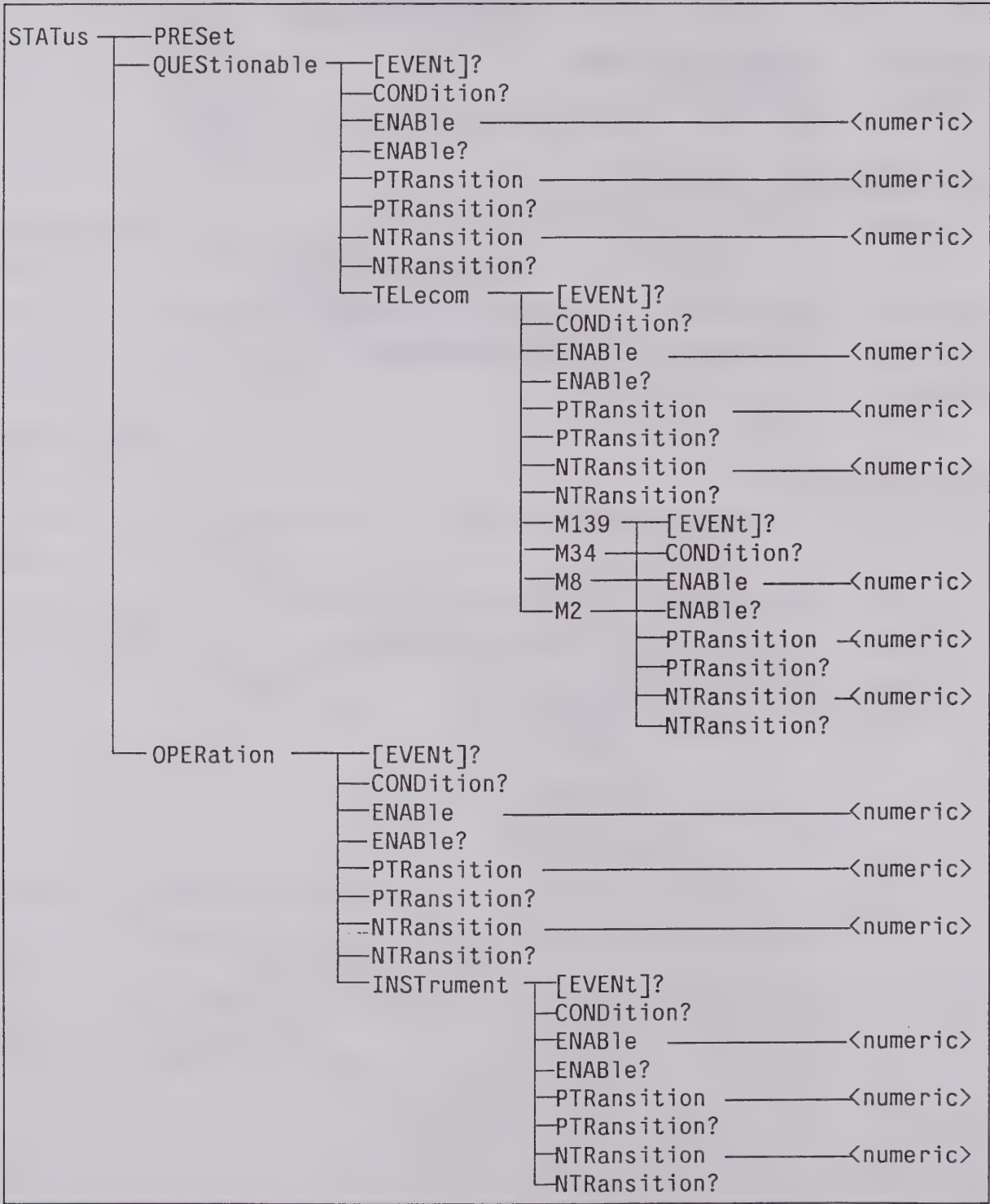
- Response <string>= <STRING RESPONSE DATA>
 "XNNN" X : One alphabetical character expressing a self-test item
 NNN: Three-digit numeric value expressing the details of an error (octal number)



- Function Queries the self-test results.
 Two or more responses are returned by separating them with commas.
 Example "A001, B001, ..."
- Example > :TEST:DATA?
 < "A001,B001,C001"

9.3.9 STATus subsystem

The STATus subsystem is used to control (set, display, etc.) Status Registers.



■ :STATus:PRESet

- Parameter None
- Function Initializes status registers.

This command affects only the following registers:
 - ▶ Transition and Enable Registers which are the status registers defined by the SCPI.
 - ▶ Transition and Enable Registers which are used as device-dependent status registers
- Example > :STATus:PRESet

(1) <QUESTIONable Status Register>

The QUESTIONable Status Register is used to provide a summary message bit from the DATA Status register.

■ :STATus:QUESTionable[:EVENT]?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits be comes the response.
- Function Queries the contents of the Event Register in the Questionable Status Register structure.
- Example > :STATus:QUESTionable:EVENT?
or :STATus:QUESTionable?
< 512

■ :STATus:QUESTionable:CONDition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits be comes the response.
- Function Queries the contents of the Condition Register in the Questionable Status Register structure.
- Example > :STATus:QUESTionable:CONDition?
< 512

■ :STATus:QUESTionable:ENABle <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

512 (Bit 9) TELecom Status Register summary

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.
- Function Sets the Event Enable Register mask. The Event Register status corresponding to mask is reported in the QUESTionable summary bit.

If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE QUESTionable summary message bit.
- Example Enable bit 9 in the Register.
> :STATus:QUESTionable:ENABle 512

■ :STATUs:QUEStionable:ENABle?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits be comes the response.
- Function Queries the mask which has been set in the Event Enable Register.
- Example >:STATUs:QUEStionable:ENABle?
 < 512

■ :STATUs:QUEStionable:PTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 512 (Bit 9) TELecom Status Register summary
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Positive Transition Filter.
 When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding QUEStionable Condition Register bit allows a 1 to be written in the corresponding QUEStionable Event Register bit.
- Example Set bit 9 in the Register.
 >:STATUs:QUEStionable:PTRansition 512

■ :STATUs:QUEStionable:PTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits be comes the response.
- Function Queries the settings which have been set in the Positive Transition Register.
- Example >:STATUs:QUEStionable:PTRansition?
 < 512

■ :STATUs:QUEStionable:NTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 512 (Bit 9) TELecom Status Register summary
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Negative Transition Filter.
 When a bit has been set to 1 in the Negative Transition Filter, a 1 to 0 transition in the corresponding QUEStionable Condition Register bit allows a 1 to be written in the corresponding QUEStionable Event Register bit.
- Example Set bit 9 in the Register.
 >:STATUs:QUEStionable:NTRansition 512

■ :STATUS:QUESTIONable:NTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Negative Transition Register.
- Example >:STATUS:QUESTIONable:NTRansition?
 < 512

(2) <TELEcom Status Register>

The TELEcom Status Register is used to provide a summary message bit from the 139M/34M/8M/2M Status registers and to display power-off status etc

■ :STATUS:QUESTIONable:TELEcom[:EVENT]?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Event Register in the TELEcom Status Register structure.
- Example >:STATUS:QUESTIONable:TELEcom:EVENT?
 or :STATUS:QUESTIONable:TELEcom?
 < 2050 (Indicates that bits 1 and 11 have been set)

■ :STATUS:QUESTIONable:TELEcom:CONDition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Condition Register in the TELEcom Status Register structure.
- Example >:STATUS:QUESTIONable:TELEcom:CONDition?
 < 4100 (Indicates that bits 2 and 12 have been set)

■ :STATUS:QUESTIONable:TELEcom:ENABle <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
- | | | |
|-------|----------|------------------------------|
| 1 | (Bit 0) | Code error occurrence |
| 2 | (Bit 1) | Bit error occurrence |
| 4 | (Bit 2) | Pattern sync occurrence |
| 8 | (Bit 3) | Signal loss occurrence |
| 16 | (Bit 4) | Power-fail occurrence |
| 32 | (Bit 5) | Hit count is counted |
| 64 | (Bit 6) | UNLOCK occurrence |
| 2048 | (Bit 11) | 2M Status Register summary |
| 4096 | (Bit 12) | 8M Status Register summary |
| 8192 | (Bit 13) | 34M Status Register summary |
| 16384 | (Bit 14) | 139M Status Register summary |

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- **Function** Sets the Event Enable Register mask. The Event Register status corresponding to mask is reported in the TELEcom summary

 If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE TELEcom summary message bit.
- **Example** Set bits 4 and 14 in the Register.

 >:STATus:QUESTionable:TELEcom:ENABle 16400

■ :STATus:QUESTionable:TELEcom:ENABle?

- **Response** <NUMERIC> = <NR1 NUMERIC RESPONSE DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.
- **Function** Queries the mask settings which have been set in the Event Enable Register.
- **Example** >:STATus:QUESTionable:TELEcom:ENABle?
 < 16400

■ :STATus:QUESTionable:TELEcom:PTRansition <numeric>

- **Parameter** <numeric> = <DECIMAL NUMERIC PROGRAM DATA>

 1 (Bit 0) Code error occurrence
 2 (Bit 1) Bit error occurrence
 4 (Bit 2) Pattern sync occurrence
 8 (Bit 3) Signal loss occurrence
 16 (Bit 4) Power-fail occurrence
 32 (Bit 5) Hit count is counted
 64 (Bit 6) UNLOCK occurrence
 2048 (Bit 11) 2M Status Register summary
 4096 (Bit 12) 8M Status Register summar
 8192 (Bit 13) 34M Status Register summary
 16384 (Bit 14) 139M Status Register summary

 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- **Function** Sets the Positive Transition Filter.

 When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding TELEcom Condition Register bit allows a 1 to be written in the corresponding TELEcom Event Register bit.
- **Example** Set bits 2 and 13 in the Register.

 >:STATus:QUESTionable:TELEcom:PTRansition 8200

■ :STATus:QUESTionable:TELEcom:PTRansition?

- **Response** <numeric> = <NR1 NUMERIC RESPONSE DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.
- **Function** Queries the settings which have been set in the Positive Transition Filter.
- **Example** >:STATus:QUESTionable:TELEcom:PTRansition?
 < 8200

■ :STATus:QUESTionable:TELEcom:NTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	Code error occurrence
2	(Bit 1)	Bit error occurrence
4	(Bit 2)	Pattern sync occurrence
8	(Bit 3)	Signal loss occurrence
16	(Bit 4)	Power-fail occurrence
32	(Bit 5)	Hit count is counted
64	(Bit 6)	UNLOCK occurrence
2048	(Bit 11)	2M Status Register summary
4096	(Bit 12)	8M Status Register summary
8192	(Bit 13)	34M Status Register summary
16384	(Bit 14)	139M Status Register summary

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- Function Sets the Negative Transition Filter.
When a bit has been set to 1 in the Negative Transition Filter, a 1 to 0 transition in the corresponding TELEcom Condition Register bit allows a 1 to be written in the corresponding TELEcom Event Register bit.
- Example Set bits 3 and 13 in the Register.
>:STATus:QUESTionable:TELEcom:NTRansition 4104

■ :STATus:QUESTionable:TELEcom:NTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Negative Transition Filter.
- Example >:STATus:QUESTionable:TELEcom:PTRansition?
< 4104

(3) <139M Status Register>

The 139M Status Register is used to display the frame errors in the 139M signal.

■ :STATus:QUESTionable:TELEcom:M139[:EVENT]

- Parameter <numeric>= <NR1 NUMERIC PROGRAM DATA>
The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Event Register in the 139M Status Register structure.
- Example >:STATus:QUESTionable:TELEcom:M139:EVENT?
or :STATus:QUESTionable:TELEcom:M139?
<3 (Indicates that bits 0 and 1 have been set)

■ :STATus:QUESTionable:TELEcom:M139:CONDition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Condition Register in the 139M Status Register structure.
- Example >:STATus:QUESTionable:TELEcom:M139:CONDition?
 < 6 (Indicates that bits 1 and 2 have been set)

■ :STATus:QUESTionable:TELEcom:M139:ENABle <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 (Bit 0) 139M LOF occurrence
 2 (Bit 1) 139M RA occurrence
 4 (Bit 2) 139M AIS occurrence
 16384 (Bit 14) 139M frame error occurrence
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Event Enable Register mask. The Event Register status corresponding to mask is reported in the 139M summary bit. If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE 139M summary message bit.
- Example Set bits 2 and 14 in the Register.
 >:STATus:QUESTionable:TELEcom:M139:ENABle 16388

■ :STATus:QUESTionable:TELEcom:M139:ENABle?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the mask settings which have been set in the Event Enable Register.
- Example >:STATus:QUESTionable:TELEcom:M139:ENABle?
 < 16388

■ :STATus:QUESTionable:TELEcom:M139:PTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 1 (Bit 0) 139M LOF occurrence
 2 (Bit 1) 139M RA occurrence
 4 (Bit 2) 139M AIS occurrence
 16384 (Bit 14) 139M frame error occurrence
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Positive Transition Filter.
 When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding 139M Condition Register bit allows a 1 to be written in the corresponding 139M Event Register bit.

- Example Set bits 0 and 14 in the Register.

```
> :STATus:QUESTionable:TELEcom:M139:PTRansition 16385
```

■ :STATus:QUESTionable:TELEcom:M139:PTRansition?

- Response <NUMERIC>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the settings which have been set in the Positive Transition Filter.

- Example > :STATus:QUESTionable:TELEcom:M139:PTRansition?
< 16385

■ :STATus:QUESTionable:TELEcom:M139:NTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	139M LOF occurrence
2	(Bit 1)	139M RA occurrence
4	(Bit 2)	139M AIS occurrence
16384	(Bit 14)	139M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- Function Sets the Negative Transition Filter.

When a bit has been set to 1 in the Negative Transition Filter, a 1 to 0 transition in the corresponding 139M Condition Register bit allows a 1 to be written in the corresponding 139M Event Register bit.

- Example Set bit 1 in the Register.

```
> :STATus:QUESTionable:TELEcom:M139:NTRansition 2
```

■ :STATus:QUESTionable:TELEcom:M139:NTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the settings which have been set in the Negative Transition Filter.

- Example > :STATus:QUESTionable:TELEcom:M139:NTRansition?
< 2

(4) <34M Status Register>

The 34M Status Register is used to display the frame errors in the 34M signal.

■ :STATus:QUESTionable:TELEcom:M34[:EVENT]?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the contents of the Event Register in the 34M Status Register structure.

- Example `>:STATus:QUESTionable:TELecom:M34:EVENT?`
 or `>:STATus:QUESTionable:TELecom:M34?`
 `<3` (Indicates that bits 0 and 1 have been set)

■ `:STATus:QUESTionable:TELecom:M34:CONDition?`

- Response `<numeric>= <NR1 NUMERIC RESPONSE DATA>`
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Condition Register in the 34M Status Register structure.
- Example `>:STATus:QUESTionable:TELecom:M34:CONDition?`
 `<6` (Indicates that bits 1 and 2 have been set)

■ `:STATus:QUESTionable:TELecom:M34:ENABle <numeric>`

- Parameter `<numeric>= <DECIMAL NUMERIC PROGRAM DATA>`

1	(Bit 0)	34M LOF occurrence
2	(Bit 1)	34M RA occurrence
4	(Bit 2)	34M AIS occurrence
16384	(Bit 14)	34M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Event Enable Register mask. The Event Register status corresponding to mask is reported in the 34M summary bit.

If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE 34M summary message bit.
- Example Set bits 2 and 14 in the Register.
 `>:STATus:QUESTionable:TELecom:M34:ENABle 16388`

■ `:STATus:QUESTionable:TELecom:M34:ENABle?`

- Response `<numeric>= <NR1 NUMERIC RESPONSE DATA>`
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Event Enable Register.
- Example `>:STATus:QUESTionable:TELecom:M34:ENABle?`
 `<16388`

■ `:STATus:QUESTionable:TELecom:M34:PTRansition <numeric>`

- Parameter `<numeric>= <DECIMAL NUMERIC PROGRAM DATA>`

1	(Bit 0)	34M LOF occurrence
2	(Bit 1)	34M RA occurrence
4	(Bit 2)	34M AIS occurrence
16384	(Bit 14)	34M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.

- **Function** Sets the Positive Transition Filter.
 When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding 34M Condition Register bit allows a 1 to be written in the corresponding 34M Event Register bit.
- **Example** Set bits 0 and 14 in the Register.
 >:STATUS:QUESTionable:TELEcom:M34:PTRansition 16385

■ :STATUS:QUESTionable:TELEcom:M34:PTRansition?

- **Response** <numeric>=<NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- **Function** Queries the settings which have been set in the Positive Transition Filter.
- **Example** >:STATUS:QUESTionable:TELEcom:M34:PTRansition?
 < 16385

■ :STATUS:QUESTionable:TELEcom:M34:NTRansition <numeric>

- **Parameter** <numeric>=<DECIMAL NUMERIC PROGRAM DATA>
 1 (Bit 0) 34M LOF occurrence
 2 (Bit 1) 34M RA occurrence
 4 (Bit 2) 34M AIS occurrence
 16384 (Bit 14) 34M frame error occurrence
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- **Function** Sets the Negative Transition Filter.
 When a bit has been set to 1 in the Negative Transition Filter, a 1 to 0 transition in the corresponding 34M condition Register bit allows a 1 to be written in the corresponding 34M Event Register bit.
- **Example** Set bits 0 and 2 in the Register.
 >:STATUS:QUESTionable:TELEcom:M34:NTRansition 5

■ :STATUS:QUESTionable:TELEcom:M34:NTRansition?

- **Response** <numeric>=<NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- **Function** Queries the settings which have been set in the Negative Transition Filter.
- **Example** >:STATUS:QUESTionable:TELEcom:M34:NTRansition?
 < 5

(5) <8M Status Register>

The 8M Status Register is used to display the frame errors in the 8M signal.

■ :STATUs:QUESTionable:TELEcom:M8[:EVENT]?

- Response <numeric>= <NR1 NUMERIC PROGRAM DATA>
The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Event Register in the 8M Status Register structure.
- Example >:STATUs:QUESTionable:TELEcom:M8:EVENT?
or :STATUs:QUESTionable:TELEcom:M8?
< 1 (Indicates that bit 0 has been set)

■ :STATUs:QUESTionable:TELEcom:M8:CONDition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Condition Register in the 8M Status Register structure.
- Example >:STATUs:QUESTionable:TELEcom:M8:CONDition?
< 2 (Indicates that bit 1 has been set)

■ :STATUs:QUESTionable:TELEcom:M8:ENABle <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	8M LOF occurrence
2	(Bit 1)	8M RA occurrence
4	(Bit 2)	8M AIS occurrence
16384	(Bit 14)	8M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.
Always set 0 to make all bits FALSE.
- Function Sets the Event Register mask. The Event Register status corresponding to mask is reported in the 8M summary bit.
If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE 8M summary message bit.
- Example Set bit 2 in the Register.
>:STATUs:QUESTionable:TELEcom:M8:ENABle 4

■ :STATUs:QUESTionable:TELEcom:M8:ENABle?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Event Enable Register.
- Example >:STATUs:QUESTionable:TELEcom:M8:ENABle?
< 4

■ :STATus:QUESTionable:TELEcom:M8:PTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	8M LOF occurrence
2	(Bit 1)	8M RA occurrence
4	(Bit 2)	8M AIS occurrence
16384	(Bit 14)	8M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- Function Sets the Positive Transition Filter.
When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding 8 M Condition Register bit allows a 1 to be written in the corresponding 8 M Event Register bit.
- Example Set bit 14 in the Register.
>:STATus:QUESTionable:TELEcom:M8:PTRansition 16384

■ :STATus:QUESTionable:TELEcom:M8:PTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the settings which have been set in the Positive Transition Filter.
- Example >:STATus:QUESTionable:TELEcom:M8:PTRansition?
< 16384

■ :STATus:QUESTionable:TELEcom:M8:NTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	8M LOF occurrence
2	(Bit 1)	8M RA occurrence
4	(Bit 2)	8M AIS occurrence
16384	(Bit 14)	8M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- Function Sets the Negative Transition Filter.
When a bit has been set to 1 in the Negative Transition Filter, a 1 to 0 transition in the corresponding 8M Condition Register bit allows a 1 to be written in the corresponding 8M Event Register bit.
- Example Set bits 0 and 14 in the Register.
>:STATus:QUESTionable:TELEcom:M8:NTRansition 16385

■ :STATus:QUESTionable:TELEcom:M8:NTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the settings which have been set in the Negative Transition Filter.

- Example > :STATus:QUESTionable:TELEcom:M8:NTRansition?
 < 16385

(6) <2M Status Register>

The 2M Status Register is used to display the frame errors in the 2M signal.

■ :STATus:QUESTionable:TELEcom:M2[:EVENT]?

- Response <numeric>= <NR1 NUMERIC PROGRAM DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Event Register in the 2M Status Register structure.
- Example > :STATus:QUESTionable:TELEcom:M2:EVENT?
 < 24 (Indicates that bits 3 and 4 have been set)

■ :STATus:QUESTionable:TELEcom:M2:CONDition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Condition Register in the k2M Status Register structure.
- Example > :STATus:QUESTionable:TELEcom:M2:CONDition?
 < 8200 (Indicates that bits 3 and 13 have been set)

■ :STATus:QUESTionable:TELEcom:M2:ENABle <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

 1 (Bit 0) 2M LOF occurrence
 2 (Bit 1) 2M RA occurrence
 4 (Bit 2) 2M AIS occurrence
 8 (Bit 3) LOG MF occurrence
 16 (Bit 4) RA MF occurrence
 8192 (Bit 13) CRC error occurrence
 16384 (Bit 14) 2M frame error occurrence

 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Event Enable Register mask. The Event Register status corresponding to mask is reported in the 2M summary bit. If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE 2M summary message bit.
- Example Set bits 13 and 145 in the Register.
 > :STATus:QUESTionable:TELEcom:M2:ENABle 24576

■ :STATus:QUESTionable:TELEcom:M2:ENABle?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the settings which have been set in the Event Enable Register.
- Example

```
>:STATus:QUESTionable:TELEcom:M2:ENABle?
< 24576
```

■ :STATus:QUESTionable:TELEcom:M2:PTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	2M LOF occurrence
2	(Bit 1)	2M RA occurrence
4	(Bit 2)	2M AIS occurrence
8	(Bit 3)	LOG MF occurrence
16	(Bit 4)	RA MF occurrence
8192	(Bit 13)	CRC error occurrence
16384	(Bit 14)	2M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- Function Sets the Positive Transition Filter.
When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding 2M Condition Register bit allows a 1 to be written in the corresponding 2M Event Register bit.
- Example Set bits 3 and 4 in the Register.

```
>:STATus:QUESTionable:TELEcom:M2:PTRansition 24
```

■ :STATus:QUESTionable:TELEcom:M2:PTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the settings which have been set in the Positive Transition filter
- Example

```
>:STATus:QUESTionable:TELEcom:M2:PTRansition?
< 24
```

■ :STATus:QUESTionable:TELEcom:M2:NTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	2M LOF occurrence
2	(Bit 1)	2M RA occurrence
4	(Bit 2)	2M AIS occurrence
8	(Bit 3)	LOG MF occurrence
16	(Bit 4)	RA MF occurrence
8192	(Bit 13)	CRC error occurrence
16384	(Bit 14)	2M frame error occurrence

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- Function Sets the Negative Transition Filter. When a bit has been set to 1 in the Negative Transition Filter, a 1 to 0 transition in the corresponding 2M Condition Register bit allows a 1 to be written in the corresponding 2M Event Register bit.

- Example Set bits 4 and 14 in the Register.
 >:STATus:QUESTionable:TELecom:M2:NTRansition 16400

■ :STATus:QUESTionable:TELecom:M2:NTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Negative Transition Filter.
- Example >:STATus:QUESTionable:TELecom:M2:NTRansition?
 < 16400

(7) <OPERation Status Register>

The OPERation Status Register is used to provide a summary message bit from the Instrument Status Register, and displays that measurement execution is in progress on this instrument.

■ :STATus:OPERation[:EVENT]?

- Parameter <numeric>= <NR1 NUMERIC PROGRAM DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Event Register in the OPERation Status Register structure.
- Example >:STATus:OPERation:EVENT?
 or :STATus:OPERation?
 < 16 (Indicates that bit 4 has been set)

■ :STATus:OPERation:CONDition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Condition Register in the OPERation Status Register structure.
- Example >:STATus:OPERation:CONDition?
 < 8192 (Indicates that bit 13 has been set)

■ :STATus:QUESTionable:ENABle <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 16 (Bit 4) Measurement execution in progress
 8192 (Bit 13) INSTRument Status Register summary
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Event Enable Register mask. The Event Register status corresponding to mask is reported in the OPERation summary bit.
 If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE OPERation summary message bit.

- Example Set bit 4 in the Register.
 >:STATus:OPERation:ENABle 16

■ :STATus:OPERation:ENABle?

- Response <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Event Enable Register
- Example >:STATus:OPERation:ENABle?
 < 16

■ :STATus:OPERation:PTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 16 (Bit 4) Measurement execution in progress
 8192 (Bit 13) INSTRument Status Register summary
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Positive Transition Filter.
 When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding OPERATION Condition Register bit allows a 1 to be written in the corresponding OPERATION Event Register bit.
- Example Set bit 13 in the Register.
 >:STATus:OPERation:PTRansition 8192

■ :STATus:OPERation:PTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>
 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Positive Transition Filter.
- Example >:STATus:OPERation:PTRansition?
 < 8192

■ :STATus:OPERation:NTRansition <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>
 16 (Bit 4) Measurement execution in progress
 8192 (Bit 13) INSTRument Status Register summary
 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.
- Function Sets the Negative Transition Filter.
 When a bit has been set to 1 in the Negative Transition Filter.
 A 1 to 0 transition in the corresponding OPERATION Condition Register bit allows a 1 to be written in the corresponding OPERATION Event Register bit.

- Example Set bits 4 and 13 in the Register.
 >:STATus:OPERation:NTRansition 8208

■ :STATus:OPERation:NTRansition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the settings which have been set in the Negative Transition Filter.
- Example >:STATus:OPERation:NTRansition?
 < 8208

(8) <INSTRUMENT Status Register>

The Instrument Status Register is used to display self-test completion, log information, etc.

■ :STATus:OPERation:INSTRument[:EVENT]?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Event Register in the INSTRument Status Register structure.
- Example >:STATus:OPERation:INSTRument:EVENT?
 or :STATus:OPERation:INSTRument?
 < 3 (Indicates that bits 1 and 2 have been set)

■ :STATus:OPERation:INSTRument:CONDition?

- Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

 The sum of the bit digit values corresponding to the set bits becomes the response.
- Function Queries the contents of the Condition Register in the INSTRument Status Register structure.
- Example >:STATus:OPERation:INSTRument:CONDition?
 < 6 (Indicates that bits 1 and 2 have been set)

■ :STATus:OPERation:INSTRument:ENABLE <numeric>

- Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

 1 (Bit 0) Log full
 2 (Bit 1) Log empty
 4 (Bit 2) Test (measurement) end
 8 (Bit 3) Self-test end
 16 (Bit 4) Change in alarm
 32 (Bit 5) Jitter measurement end

 Set the sum of the bit digit values corresponding to the bits to be set.
 Always set 0 to make all bits FALSE.

- **Function** Sets the Event Enable Register mask. The Event Register status corresponding to mask is reported in the INSTRument summary bit.

If a bit is set to 1 in the Event Enable Register, the corresponding event bit transitions to TRUE will cause a TRUE INSTRument summary message bit.

- **Example** Set bits 2 and 3 in the Register.

```
>:STATus:OPERation:INSTRument:ENABle 12
```

■ :STATus:OPERation:INSTRument:ENABle?

- **Response** <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- **Function** Queries the settings which have been set in the Event Enable Register.

- **Example** >:STATus:OPERation:INSTRument:ENABle?
< 12

■ :STATus:OPERation:INSTRument:PTRansition <numeric>

- **Parameter** <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	Log full
2	(Bit 1)	Log empty
4	(Bit 2)	Test (measurement) end
8	(Bit 3)	Self-test end
16	(Bit 4)	Change in alarm
32	(Bit 5)	Jitter measurement end

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- **Function** Sets the Positive Transition Filter.

When a bit has been set to 1 in the Positive Transition Filter, a 0 to 1 transition in the corresponding INSTRument Condition Register bit allows a 1 to be written in the corresponding INSTRument Event Register bit.

- **Example** Set bits 3 and 4 in the Register.

```
>:STATus:OPERation:INSTRument:PTRansition 24
```

■ :STATus:OPERation:INSTRument:PTRansition?

- **Response** <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- **Function** Queries the settings which have been set in the Positive Transition Filter.

- **Example** >:STATus:OPERation:INSTRument:PTRansition?
< 24

■ :STATus:OPERation:INSTRument:NTRansition <numeric>

● Parameter <numeric>= <DECIMAL NUMERIC PROGRAM DATA>

1	(Bit 0)	Log full
2	(Bit 1)	Log empty
4	(Bit 2)	Test (measurement) end
8	(Bit 3)	Self-test end
16	(Bit 4)	Change in alarm
32	(Bit 5)	Jitter measurement end

Set the sum of the bit digit values corresponding to the bits to be set.

Always set 0 to make all bits FALSE.

- Function Sets the Negative Transition Filter.

When a bit has been set to 1 in the Negative Transition Filter, a 1 to 0 transition in the corresponding INSTRument Condition Register bit allows a 1 to be written in the corresponding INSTRument Event Register bit.
- Example Set bits 0 and 3 in the Register.

>:STATus:OPERation:INSTRument:NTRansition 9

■ :STATus:OPERation:INSTRument:NTRansition?

● Response <numeric>= <NR1 NUMERIC RESPONSE DATA>

The sum of the bit digit values corresponding to the set bits becomes the response.

- Function Queries the settings which have been set in the Negative Transition Filter.
- Example >:STATus:OPERation:INSTRument:NTRansition?
 < 9

SECTION 10

PROGRAM EXAMPLES

This section describes examples of programs for executing simplified measurements using the remote control options. The programs explained here are created using Quick BASIC.

The programs explained in the following pages can apply to both the GPIB and the RS-232C interfaces. Blocks in which different settings are required because of the interface type or controller type, are programed using subroutines. A subroutine should be created according to the circumstances in which a program is created.

TABLE OF CONTENTS

10.1	Loop Back Measurement	10-3
10.2	Monitor Measurements	10-10

(Blank)

SECTION 10

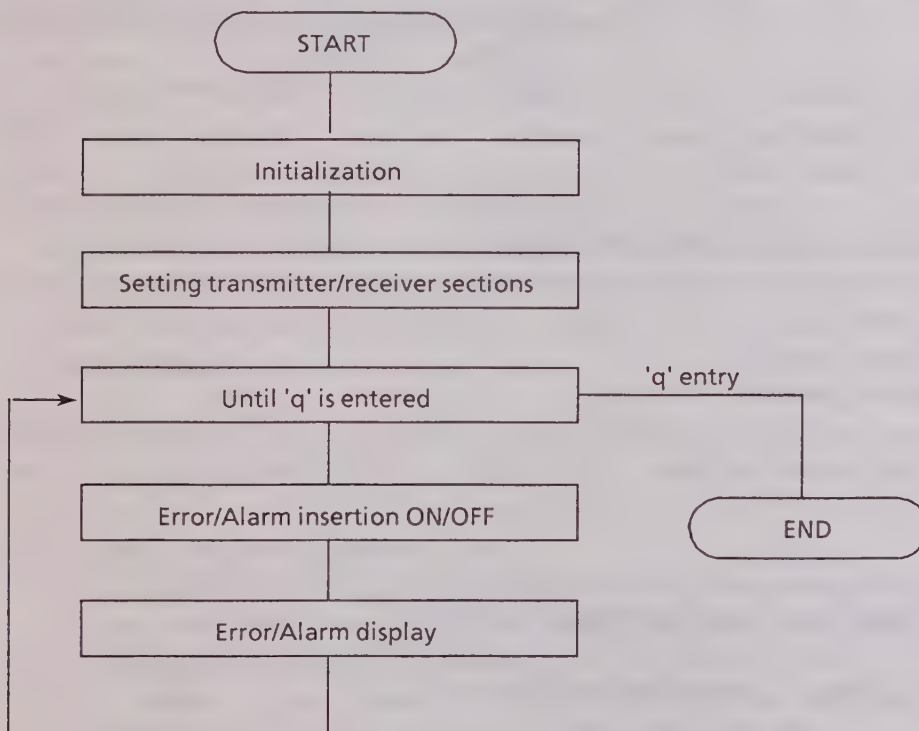
PROGRAM EXAMPLES

10.1 Loop Back Measurement

Loop back measurements are performed in out-of service mode. Measurements are repeated every 5 seconds. The measured results of the bit errors generated during a single measurement, AIS 139M, and RA 139M, are displaced.

The program is also designed so that error/alarm insertion can be switched ON or OFF from the keyboard. Program execution continues until a 'q' character is entered from the keyboard.

The diagram below shows a simplified program flowchart.



Sample Program (1) is shown on the following page.

MP1520B PDH Analyzer

Sample Program (1)

' Header file included

' The constants to be used are defined in the head.bas' file included
' in this program. They are defined as follows.

' CONST SON%=1 Indicates the ON state.

' CONST SOFF%=0 Indicates the OFF state.

' CONST ZERO\$=" 0" Used when deciding whether or not the
' measured result is 0.

' CONST EOT%=4 Used when deciding whether bit 2 in the
' INSTRument Status Event Register is set
' or not. (4 = 2²)

' CONST TRUE%=-1 Indicates the TRUE state.

REM \$INCLUDE: 'head.bas'

' Declaring subroutines and functions

DECLARE SUB init ()

DECLARE SUB send (cmd\$)

DECLARE SUB rcv (resp\$)

' Initializing interface etc. (See init subroutine.)

CALL init

' Setting transmitter and receiver sections

CALL send (":SENSe:TELEcom:MMODE OSERvice")

'Out-of Service

CALL send (":INSTRument:COUPLE ALL")

'Settings common to transmitter and receiver sections

CALL send (":SOURCE:TELEcom:BRATe M139;PATTeRn PRBS23")

'Bit rate:139Mb/s Patt:PRBS23

CALL send (":SOURCE:TELEcom:FRAME ON") 'Frame:ON

CALL send (":SOURCE:TELEcom:ERRor:TYPE OFF") 'ERRor:OFF

CALL send (":SOURCE:TELEcom:ALARm:TYPE OFF") 'Alarm:OFF

CALL send (":SENSe:MEASure:TYPE REPeat;PERiod 5,s")

'Mode:Repeat [5 secs]

```

'-----
' Setting the initial value of the variable
'-----
char$=""                'char$: Variable used to save the key pressed
'-----
' Measurement start
'-----
CALL send (":STATus:PRESet")          'Resets status registers.
CALL send (":SENSe:MEASure:StARt")    'Measurement start
PRINT "Measurement Start"
'-----
' Displaying the measured results executed every 5 seconds (Bit error,
' AIS 139M, and RA 139M)
' The measured results are displayed every 5 seconds until a 'q'
' character is entered from the keyboard.
'-----
WHILE TRUE%
    CALL send("CLS")                'Clears event registers.
    '-----
    ' Measurement end decision and character entry
    ' Until a single measurement has been completed, decides that
    ' measurements have been completed, and switches the error/alarm
    ' insertion to ON or OFF when a specified character is entered from
    ' the keyboard.
    '-----
mend% = SOFF%                'mend%: Variable for deciding measurement end
WHILE mend% = SOFF%
    '-----
    ' Deciding measurement end
    ' To decide that a single measurement has been completed, read
    ' out the INSTRument Status Event register to check whether bit 2
    ' (measurement end decision bit) is set or not. If bit 2 is set,
    ' a number other than 0, resulting from the AND operation, is
    ' assigned to the variable mend%.
    '-----

```

```

CALL send(":STATus:OPERation:INSTRument:EVENT?")
      'Reads the value of the Event Register.
CALL rcv(st$)
mend% = VAL(st$) AND EOT%           'Checks the state of bit 2.
'-----
' Keyboard entry
' Switches the error/alarm insertion to ON or OFF, or stops
' executing the program when a specified character is entered from
' the keyboard.
' The key characters corresponding to the functions are as
' follows:
'   ." e" :           Inserts bit errors in a 139M signal at the rate
'                   of 1E-3.
'   ." a" :           Inserts AIS in a 139M signal.
'   ." r" :           Inserts RA in a 139M signal
'   ." s" :           Switches error/alarm insertion to OFF.
'   ." q" :           Stops program execution.
'-----
char$ = INKEY$
SELECT CASE char$                   'Determines a key
  CASE "e"
    CALL send ("SOURce:TELEcom:ERRor:TYPE BIT139;ERATe E_3")
  CASE "a"
    CALL send ("SOURce:TELEcom:ALARm:TYPE AIS139")
  CASE "r"
    CALL send ("SOURce:TELEcom:ALARm:TYPE RA139")
  CASE "s"
    CALL send ("SOURce:TELEcom:ERRor:TYPE OFF")
    CALL send ("SOURce:TELEcom:ALARm:TYPE OFF")
  CASE "q"
    PRINT "Measurement Stop"
    STOP
END SELECT
WEND

```

```

'-----
' Reading the time and measured results.
' Reads the time and the measured results of bit errors, AIS139M,
' and RA139M.
'-----
CALL send(":SYSTem:DATE?:TIME?")
CALL rcv(dat$)           'Queries date and time and reads them
                        in variable.
CALL send(":CALCulate:DATA? 'LAST:EC:BIT'")
CALL rcv(mdBERR$)       'Queries the bit error count and reads
                        it in variable.
CALL send(":CALCulate:DATA? 'LAST:ASEConds:AIS:M139'")
CALL rcv(mdAIS$)        'Queries the alarm second of AIS139M
                        and reads it in variable.
CALL send(":CALCulate:DATA? 'LAST:ASEConds:RA:M139'")
CALL rcv(mdRAS$)        'Queries the alarm second of RA 139M
                        and reads it in variable.
'-----
' Displaying the measured result
' Displays the error count/alarm second of the error/alarm
' which has occurred if the measured result is not 0 (when error
' /alarm has occurred.)
'-----
IF (mdBERR$ <> ZERO$) OR (mdAIS$ <> ZERO$) OR (mdRAS$ <> ZERO$) THEN
    PRINT dat$
    IF mdBERR$ <> ZERO$ THEN
        PRINT "      Bit Err", mdBERR$
    END IF
    IF mdAIS$ <> ZERO$ THEN
        PRINT "      AIS139M", mdAIS$
    END IF
    IF mdRAS$ <> ZERO$ THEN
        PRINT "      RA 139M", mdRAS$
    END IF
END IF
WEND
END

```

```

'-----
' Initializing interface etc.
'   Create a processing subroutine as shown below to suit your own
'   controller or interface.
'   GPIB interface:      Create a subroutine for executing bus
'                         initialization (IFC), message exchange
'                         initialization (DCL, SDC), and terminator
'                         settings.
'   RS-232C interface:   If the data transmission speed, parity,
'                         transmission character, etc. must be set to
'                         a controller, create subroutine for the
'                         required settings.
'-----

```

```
SUB init
```

```

:
:

```

```
END SUB
```

```

'-----
' send(cmd$) Sending commands to measuring instruments
'   Argument (cmd$): String variable for expressing the commands to
'   send.
'   Create a subroutine for sending commands to measuring instrument
'   to suit your own controller.
'-----

```

```
SUB send (cmd$)
```

```

:
:

```

```
END SUB
```

```

'-----
' rcv(resp$) Reading the data from the measuring instruments
'   Argument (resp$): String variable for assigning the values
'   returned from the measuring instruments.
'   Create a subroutine for assigning the values returned from the
'   measuring instruments to suit your own controller.
'-----

```

```
SUB rcv (resp$)
```

```

:
:

```

```
END SUB
```

<Measured result>

Measurement Start

1994,02,04;13,06,56

Bit Err 9.82E+07

AIS139M 4

RA 139M 4

1994,02,04;13,07,01

Bit Err 1.45E+08

AIS139M 5

RA 139M 5

1994,02,04;13,07,06

Bit Err 1.43E+08

AIS139M 5

RA 139M 5

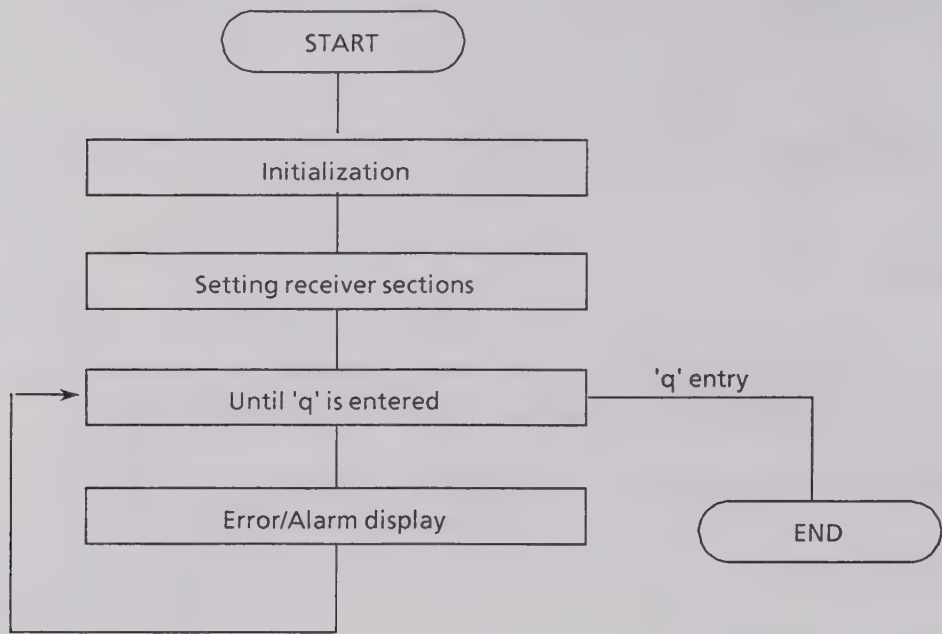
Measurement Stop

10.2 Monitor Measurements

Monitor measurements are performed in in-service mode.

Measurements are repeated every 5 seconds. The measured results of the 34M frame errors generated during single measurement, AIS 34M, and RA 34M, are displayed. Program execution continues until a 'q' character is entered from the keyboard.

The diagram below shows a simplified program flowchart.



Sample Program (2) is shown on the following page. For a detailed explanation of the same processing as for loop back measurements, see Sample Program (1) in paragraph 10.1.

```

'-----
'                               MP1520B PDH Analyzer
'                               Sample Program (2)
'-----

'-----
' Header file included (See Sample Program (1))
'     CONST SON%=1
'     CONST SOFF%=1
'     CONST ZERO$="          0"
'     CONST EOT%=4
'     CONST TRUE%=-1
'-----

REM $INCLUDE: 'head.bas'
'-----
' Declaring subroutines and functions
'-----

DECLARE SUB init ( )
DECLARE SUB send (cmd$)
DECLARE SUB rcv (resp$)
'-----
' Initializing interface etc. (See Sample Program (1) init subroutine.)
'-----

CALL init
'-----
' Setting receiver section
'-----

CALL send (":SENSe:TELEcom:MMODE ISERvice")      'In-Service
CALL send (":INSTrument:COUPle NONE")           'Different settings
                                                for transmitter and
                                                receiver section

CALL send (":SENSe:TELEcom:BRATe M34")          'Bit rate:34Mb/s
CALL send (":SENSe:TELEcom:FRAMe ON")           'Frame:ON
CALL send (":SENSe:MEASure:TYPE REPeat;PERiod 5,s")
                                                'Mode:Repeat [5 secs]
'-----
' Setting the initial value of the variable
'-----

char$=""          'char$: Variable used to save the key pressed
'-----
' Measurement start
'-----

CALL send (":STATus:PRESet")                    'Resets status registers.
CALL send (":SENSe:MEASure:START")              'Measurement start
PRINT "Measurement Start"

```

```

'-----
' Displaying the measured results executed every 5 seconds
' (34M frame error, AIS 139M, and RA 139M)
' The measured results are displayed every 5 seconds until a 'q'
' character is entered from the keyboard.
'-----
WHILE TRUE%
    CALL send("*CLS")          'Clears event registers.
    '-----
    ' Measurement end decision and keyboard entry
    '   Until a single measurement has been completed, decides that
    '   measurements have been completed, and enter a specified
    '   character from the keyboard.
    '-----
    mend% = SOFF%              'Variable for deciding measurement end
    WHILE mend% = SOFF%
        '-----
        ' Deciding measurement end
        '   To decide that a single measurement has completed, read out
        '   the INSTRUMENT Status Event Register to check whether bit 2
        '   (measurement end decision bit) is set or not. If bit 2 is
        '   set, a number other than 0, resulting from the AND
        '   operation, is assigned to the variable mend%.
        '-----
        CALL send(":STATus:OPERation:INSTRument:EVENT?")
                                'Queries the value of the Event
                                Register and reads it in variable.

        CALL rcv(st$)
        mend% = VAL(st$) AND EOT% 'Checks the state of bit 2.
        '-----
        ' Keyboard entry
        '   Stops executing the program when a 'q' character is entered from
        '   the keyboard.
        '-----
        char$ = INKEY$
        IF char$="q" THEN
            PRINT "Measurement Stop"
            STOP
        END IF
    END WHILE
END WHILE

```

```

WEND
'-----
' Reading the time and measured result.
' Reads the time and the measured results of the 34M frame error,
' AIS34M, and RA34M.
'-----
CALL send(":SYSTem:DATE?;TIME?")
CALL rcv(dat$)                'Queries date and time and reads them
                              in variable.
CALL send(":CALCulate:DATA? 'LAST:EC:FRAMe:M34'")
CALL rcv(mdBERR$)             'Queries the bit error count and reads
                              it in variable.
CALL send(":CALCulate:DATA? 'LAST:ASEConds:AIS:M34'")
CALL rcv(mdAIS$$)             'Queries the alarm second of AIS34M and
                              reads it in variable.
CALL send(":CALCulate:DATA? 'LAST:ASEConds:RA:M34'")
CALL rcv(mdRA$)               'Queries the alarm second of RA34M and
                              reads it in variable.
'-----
' Displaying the measured result
' Displays the error count/alarm second of the error/alarm which
' has occurred if the measured result is not 0 (when error/alarm
' has occurred.).
'-----
IF (mdBERR$ <> ZERO$) OR (mdAIS$ <> ZERO$) OR (mdRA$ <> ZERO$) THEN
    PRINT dat$
    IF mdBERR$ <> ZERO$ THEN
        PRINT "      FRM 34M", mdBERR$
    END IF
    IF mdAIS$ <> ZERO$ THEN
        PRINT "      AIS 34M", mdAIS$
    END IF
    IF mdRA$ <> ZERO$ THEN
        PRINT "      RA 34M", mdRA$
    END IF
END IF
WEND
END

```

```
'-----
' Initializing interface (See Sample Program (1))
'-----
```

```
SUB init
```

```
:
:
```

```
END SUB
```

```
'-----
' send(cmd$)      Sending the commands to measuring instruments
'                  (See Sample Program (1))
'-----
```

```
SUB send (cmd$)
```

```
:
:
```

```
END SUB
```

```
'-----
' rcv(resp$)      Reading the data from the measuring instruments
'                  (See Sample Program (1))
'-----
```

```
SUB rcv (resp$)
```

```
:
:
```

```
END SUB
```

<Measured result>

Measurement Start

1994,02,04;13,09,49

FRM 34M 726

1994,02,04;13,09,54

FRM 34M 30887

RA 34M 1

1994,02,04;13,09,59

FRM 34M 29486

RA 34M 5

1994,02,04;13,10,04

FRM 34M 23202

RA 34M 1

Measurement Stop

APPENDIXES

TABLE OF CONTENTS

APPENDIX A	SCPI CONFORMANCE INFORMATION	A-1
A.1	SCPI Version	A-1
A.2	SCPI-defined Commands	A-1
A.3	Non-SCPI Commands	A-2
APPENDIX B	SELF-TEST ERROR CODES	B-1
APPENDIX C	SCPI ERROR MESSAGES	C-1
C.1	Command Errors	C-1
C.2	Execution Errors	C-2
C.3	Device-Specific Errors	C-3
C.4	Query Errors	C-4
APPENDIX D	CORRESPONDENCE BETWEEN SCREENS AND COMMANDS	D-1
D.1	The Correspondence between Condtn Screens and Commands	D-1
D.1.1	Condtn:Tx&Rx screen	D-1
D.1.2	Condtn:Tx screen	D-1
D.1.3	Condtn:Rx screen	D-2
D.2	The Correspondence between Results Screens and Commands	D-2
D.3	The Correspondence between Analysis Screens and Commands	D-3
D.3.1	Analysis:Trace Text screen	D-3
D.3.2	Analysis:Trace Graph screen	D-4
D.3.3	Analysis:Distribution screen	D-4
D.4	The Correspondence between Setup Screens and Commands	D-5
D.4.1	Setup:Measure screen	D-5
D.4.2	Setup:Performance screen	D-5
D.4.3	Setup:Print screen	D-7
D.4.4	Setup:Memory screen	D-7
D.5	The Correspondence between System Screen and Commands	D-8
D.6	The Correspondence between Self-Test Screen and Commands	D-8
D.7	The Correspondence between Jitter screen and Commands	D-9
D.7.1	Jitter:Manual screen	D-9
D.7.2	Jitter:Sweep mask screen	D-10
D.7.3	Jitter:Transfer screen	D-11
D.7.4	Jitter:Jitt/Freq screen	D-12
D.8	The Correspondence between Front Panel and Other Commands	D-13

APPENDIX E	DETAILED FUNCTIONS OF INITIALIZATION COMMANDS	E-1
E.1	Bus Initialization	E-1
E.2	Message Exchange Initialization	E-2
E.3	Device Initialization	E-3
E.4	Device Status at Power-on	E-3

APPENDIX A

SCPI CONFORMANCE INFORMATION

A.1 SCPI Version

The MP1520B conforms to SCPI version 1993.0.

A.2 SCPI-defined Commands

The commands defined by SCPI 1993.0 are listed below. The commands used in this instrument are derived from these.

```
:CALCulate:DATA?
:INSTRument:COUPle
:INSTRument:COUPle?
:DISPlay:MENU[:NAME]
:DISPlay:MENU[:NAME]?
:SYSTem:DATE
:SYSTem:DATE?
:SYSTem:TIME
:SYSTem:TIME?
:SYSTem:ERRor?
:SYSTem:VERSion?
:STATus:PRESet
:STATus:QUEStionable[:EVENT]?
:STATus:QUEStionable:CONDition?
:STATus:QUEStionable:ENABle
:STATus:QUEStionable:ENABle?
:STATus:QUEStionable:PTRansition
:STATus:QUEStionable:PTRansition?
:STATus:QUEStionable:NTRansition
:STATus:QUEStionable:NTRansition?
:STATus:OPERation[:EVENT]?
:STATus:OPERation:CONDition?
:STATus:OPERation:ENABle
:STATus:OPERation:ENABle?
:STATus:OPERation:PTRansition
:STATus:OPERation:PTRansition?
:STATus:OPERation:NTRansition
:STATus:OPERation:NTRansition?
:STATus:OPERation:INSTRument[:EVENT]?
:STATus:OPERation:INSTRument:CONDition?
:STATus:OPERation:INSTRument:ENABle
:STATus:OPERation:INSTRument:ENABle?
:STATus:OPERation:INSTRument:PTRansition
:STATus:OPERation:INSTRument:PTRansition?
:STATus:OPERation:INSTRument:NTRansition
:STATus:OPERation:INSTRument:NTRansition?
```

*IDN?	*SRE
*RST	*SRE?
*TST?	*STB?
*OPC	*TRG
*OPC?	*PSC
*WAI	*PSC?
*CLS	*SAV
*ESE	*RCL
*ESE?	*OPT?
*ESR?	

A.3 Non-SCPI Commands

The SCPI-undefined commands are listed below. The commands used in this instrument are derived from these.

```
:SOURce:TELeom:BRATe
:SOURce:TELeom:BRATe?
:SOURce:TELeom:OFFSet
:SOURce:TELeom:OFFSet?
:SOURce:TELeom:MUX:MRATe
:SOURce:TELeom:MUX:MRATe?
:SOURce:TELeom:MUX:ROUTe
:SOURce:TELeom:MUX:ROUTe?
:SOURce:TELeom:MUX:N
:SOURce:TELeom:MUX:N?
:SOURce:TELeom:PATtern[:TYPE]
:SOURce:TELeom:PATtern[:TYPE]?
:SOURce:TELeom:PATtern:UWORD
:SOURce:TELeom:PATtern:UWORD?
:SOURce:TELeom:FRAMing
:SOURce:TELeom:FRAMing?
:SOURce:TELeom:ALARm:TYPE
:SOURce:TELeom:ALARm:TYPE?
:SOURce:TELeom:ALARm:ARATe
:SOURce:TELeom:ALARm:ARATe?
:SOURce:TELeom:ERRor:TYPE
:SOURce:TELeom:ERRor:TYPE?
:SOURce:TELeom:ERRor:ERATe
:SOURce:TELeom:ERRor:ERATe?
:SOURce:TELeom:JITTer
:SOURce:TELeom:JITTer?
:SOURce:JITTer:MANual:MODE
:SOURce:JITTer:MANual:MODE?
:SOURce:JITTer:MANual:RANGe
:SOURce:JITTer:MANual:RANGe?
```

```

:SOURce:JITTer:MANual:FREQuency
:SOURce:JITTer:MANual:FREQuency?
:SOURce:JITTer:MANual:AMPLitude:STEP
:SOURce:JITTer:MANual:AMPLitude:STEP?
:SOURce:JITTer:MANual:AMPLitude:DATA
:SOURce:JITTer:MANual:AMPLitude:MONiter?
:SOURce:JITTer:SMASk:TABLE:TYPE
:SOURce:JITTer:SMASk:TABLE:TYPE?
:SOURce:JITTer:SMASk:TABLE:POINT
:SOURce:JITTer:SMASk:TABLE:POINT?
:SOURce:JITTer:SMASk:TABLE:DATA
:SOURce:JITTer:SMASk:TABLE:DATA?
:SOURce:JITTer:TRANSfer:TABLE:TYPE
:SOURce:JITTer:TRANSfer:TABLE:TYPE?
:SOURce:JITTer:TRANSfer:TABLE:POINT
:SOURce:JITTer:TRANSfer:TABLE:POINT?
:SOURce:JITTer:TRANSfer:TABLE:DATA
:SOURce:JITTer:TRANSfer:TABLE:DATA?
:SOURce:JITTer:JFREquency:OFFSet:MAX
:SOURce:JITTer:JFREquency:OFFSet:MAX?
:SOURce:JITTer:JFREquency:OFFSet:MIN
:SOURce:JITTer:JFREquency:OFFSet:MIN?
:SOURce:JITTer:JFREquency:OFFSet:STEP
:SOURce:JITTer:JFREquency:OFFSet:STEP?
:SENSe:TELEcom:BRATe
:SENSe:TELEcom:BRATe?
:SENSe:TELEcom:DEMux:MRATe
:SENSe:TELEcom:DEMux:MRATe?
:SENSe:TELEcom:DEMux:ROUTe
:SENSe:TELEcom:DEMux:ROUTe?
:SENSe:TELEcom:DEMux:N
:SENSe:TELEcom:DEMux:N?
:SENSe:TELEcom:PATtern[:TYPE]
:SENSe:TELEcom:PATtern[:TYPE]?
:SENSe:TELEcom:PATtern:UWORD
:SENSe:TELEcom:PATtern:UWORD?
:SENSe:TELEcom:FRAMing
:SENSe:TELEcom:FRAMing?
:SENSe:TELEcom:MMODE
:SENSe:TELEcom:MMODE?
:SENSe:TELEcom:M2:CRC
:SENSe:TELEcom:M2:CRC?
:SENSe:TELEcom:M2:SIGNaling
:SENSe:TELEcom:M2:SIGNaling?
:SENSe:TELEcom:M2:MCHanne1

```

:SENSe:TELeCom:M2:MCHannel?
:SENSe:TELeCom:IMODE
:SENSe:TELeCom:IMODE?
:SENSe:MEASure:TYPE
:SENSe:MEASure:TYPE?
:SENSe:MEASure:PERiod
:SENSe:MEASure:PERiod?
:SENSe:MEASure:BTIme:SET
:SENSe:MEASure:BTIme:SET?
:SENSe:MEASure:BTIme:START
:SENSe:MEASure:BTIme:START?
:SENSe:MEASure:STATe?
:SENSe:MEASure:START
:SENSe:MEASure:STOP
:SENSe:MEASure:TImed?
:SENSe:MEASure:ELAPsed?
:SENSe:JITter:MANual:MMODE
:SENSe:JITter:MANual:MMODE?
:SENSe:JITter:MANual:RANGe
:SENSe:JITter:MANual:RANGe?
:SENSe:JITter:MANual:FILTer
:SENSe:JITter:MANual:FILTer?
:SENSe:JITter:MANual:THReshold
:SENSe:JITter:MANual:THReshold?
:SENSe:JITter:SMASk:START
:SENSe:JITter:SMASk:STOP
:SENSe:JITter:SMASk:STATe?
:SENSe:JITter:TRANSfer:MODE
:SENSe:JITter:TRANSfer:MODE?
:SENSe:JITter:TRANSfer:START
:SENSe:JITter:TRANSfer:STOP
:SENSe:JITter:TRANSfer:STATe?
:SENSe:JITter:JFRequency:INTerval
:SENSe:JITter:JFRequency:INTerval?
:SENSe:JITter:JFRequency:FILTer
:SENSe:JITter:JFRequency:FILTer?
:SENSe:JITter:JFRequency:STARs
:SENSe:JITter:JFRequency:STOP
:SENSe:JITter:JFRequency:STATe?
:DISPlay:RESult:TYPE
:DISPlay:RESult:TYPE?
:DISPlay:RESult:MODE
:DISPlay:RESult:MODE?
:DISPlay:RESult:ZOOM:SET
:DISPlay:RESult:ZOOM:SET?
:DISPlay:RESult:ZOOM:ITEM1
:DISPlay:RESult:ZOOM:ITEM1?
:DISPlay:RESult:ZOOM:ITEM2

```

:DISPlay:RESUlt:ZOOM:ITEM2?
:DISPlay:RESUlt:TIME
:DISPlay:RESUlt:TIME?
:DISPlay:ANALYsis:TTEXT:FROM
:DISPlay:ANALYsis:TTEXT:FROM?
:DISPlay:ANALYsis:TTEXT:FORMAt
:DISPlay:ANALYsis:TTEXT:FORMAt?
:DISPlay:ANALYsis:TTEXT:PRINt
:DISPlay:ANALYsis:TTEXT:PRINt?
:DISPlay:ANALYsis:TTEXT:SCRo11
:DISPlay:ANALYsis:TGRaph:ERRor
:DISPlay:ANALYsis:TGRaph:ERRor?
:DISPlay:ANALYsis:TGRaph:INTerval
:DISPlay:ANALYsis:TGRaph:INTerval?
:DISPlay:ANALYsis:TGRaph:PRINt
:DISPlay:ANALYsis:TGRaph:PRINt?
:DISPlay:ANALYsis:TGRaph:SCRo11
:DISPlay:ANALYsis:TGRaph:FROM
:DISPlay:ANALYsis:TGRaph:FROM?
:DISPlay:ANALYsis:TGRaph:ALARm1
:DISPlay:ANALYsis:TGRaph:ALARm1?
:DISPlay:ANALYsis:TGRaph:ALARm2
:DISPlay:ANALYsis:TGRaph:ALARm2?
:DISPlay:ANALYsis:TGRaph:ALARm3
:DISPlay:ANALYsis:TGRaph:ALARm3?
:DISPlay:ANALYsis:DISTribution:ERRor
:DISPlay:ANALYsis:DISTribution:ERRor?
:DISPlay:JITTer:MANual:MODE
:DISPlay:JITTer:MANual:MODE?
:DISPlay:JITTer:MANual:ZOOM:SET
:DISPlay:JITTer:MANual:ZOOM:SET?
:DISPlay:JITTer:MANual:ZOOM:ITEM1
:DISPlay:JITTer:MANual:ZOOM:ITEM1?
:DISPlay:JITTer:MANual:ZOOM:ITEM2
:DISPlay:JITTer:MANual:ZOOM:ITEM2?
:DISPlay:JITTer:MANual:TIME
:DISPlay:JITTer:MANual:TIME?
:DISPlay:JITTer:TRANSfer:MODE
:DISPlay:JITTer:TRANSfer:MODE?
:DISPlay:JITTer:TRANSfer:SCALe
:DISPlay:JITTer:TRANSfer:SCALe?
:DISPlay:JITTer:JFRequency:SCALe
:DISPlay:JITTer:JFRequency:SCALe?
:ROUTe:INPut:SELEct
:ROUTe:INPut:SELEct?
:CALCulate:TELEcom:PERFormance:TYPE

```

```

:CALCulate:TELEcom:PERFormance:TYPE?
:CALCulate:TELEcom:PERFormance:FRAMe
:CALCulate:TELEcom:PERFormance:FRAMe?
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:SET
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:SET
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:S1
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:S1
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:S2
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:ES:S2?
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:S2
:CALCulate:TELEcom:PERFormance:THReshoId:M2100:SES:S2?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:ES:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G821:ES:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:SES:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G821:SES:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:DM:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G821:DM:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:US:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G821:US:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:ES:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G821:ES:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:SES:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G821:SES:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:DM:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G821:DM:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:G821:US:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G821:US:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:ESR:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G826:ESR:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:SESR:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G826:SESR:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:BBER:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G826:BBER:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:US:SET
:CALCulate:TELEcom:PERFormance:THReshoId:G826:US:SET?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:ESR:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G826:ESR:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:SESR:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G826:SESR:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:BBER:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G826:BBER:S1?
:CALCulate:TELEcom:PERFormance:THReshoId:G826:US:S1
:CALCulate:TELEcom:PERFormance:THReshoId:G826:US:S1?
:CALCulate:LOG:FORMat

```

```

:CALCulate:LOG:FORMat?
:CALCulate:LOG:DATA?
:SYSTem:PRINt:IDATa[:TYPE]
:SYSTem:PRINt:IDATa[:TYPE]?
:SYSTem:PRINt:IDATa:PERiod
:SYSTem:PRINt:IDATa:PERiod?
:SYSTem:PRINt:IDATa:THReshold
:SYSTem:PRINt:IDATa:THReshold?
:SYSTem:PRINt:MCONdition
:SYSTem:PRINt:MCONdition?
:SYSTem:PRINt:ERRor:SET
:SYSTem:PRINt:ERRor:SET?
:SYSTem:PRINt:ERRor:TYPE
:SYSTem:PRINt:ERRor:TYPE?
:SYSTem:PRINt:ERRor:THReshold:EC
:SYSTem:PRINt:ERRor:THReshold:EC?
:SYSTem:PRINt:ERRor:THReshold:ER
:SYSTem:PRINt:ERRor:THReshold:ER?
:SYSTem:PRINt:ERRor:SQUelch
:SYSTem:PRINt:ERRor:SQUelch?
:SYSTem:PRINt:ALARm:SET
:SYSTem:PRINt:ALARm:SET?
:SYSTem:PRINt:MDATa:TYPE
:SYSTem:PRINt:MDATa:TYPE?
:SYSTem:PRINt:MDATa:ERRor
:SYSTem:PRINt:MDATa:ERRor?
:SYSTem:PRINt:MDATa:PERFormance
:SYSTem:PRINt:MDATa:PERFormance?
:SYSTem:PRINt:MDATa:ALARm
:SYSTem:PRINt:MDATa:ALARm?
:SYSTem:PRINt:MDATa:JITTer
:SYSTem:PRINt:MDATa:JITTer?
:SYSTem:PRINt:ENABle
:SYSTem:PRINt:ENABle?
:SYSTem:PRINt:COpy
:SYSTem:PRINt:FEED
:SYSTem:PRINt:TEXT
:SYSTem:MEMory:RECa11
:SYSTem:MEMory:STORE
:SYSTem:MEMory:CLEAr
:SYSTem:MEMory:LABel
:SYSTem:MEMory:LABel?
:SYSTem:BEEPer:TYPE
:SYSTem:BEEPer:TYPE?
:SYSTem:LED:RESet
:TEST:STARt
:TEST:STOP
:TEST:DATA?

```

```

:TEST:STATe?
:TEST:JITTer
:TEST:JITTer?
:STATus:QUESTionable:TELEcom[:EVENT]?
:STATus:QUESTionable:TELEcom:CONDition?
:STATus:QUESTionable:TELEcom:ENABle
:STATus:QUESTionable:TELEcom:ENABle?
:STATus:QUESTionable:TELEcom:PTRansition
:STATus:QUESTionable:TELEcom:PTRansition?
:STATus:QUESTionable:TELEcom:NTRansition
:STATus:QUESTionable:TELEcom:NTRansition?
:STATus:QUESTionable:TELEcom:M139[:EVENT]?
:STATus:QUESTionable:TELEcom:M139:CONDition?
:STATus:QUESTionable:TELEcom:M139:ENABle
:STATus:QUESTionable:TELEcom:M139:ENABle?
:STATus:QUESTionable:TELEcom:M139:PTRansition
:STATus:QUESTionable:TELEcom:M139:PTRansition?
:STATus:QUESTionable:TELEcom:M139:NTRansition
:STATus:QUESTionable:TELEcom:M139:NTRansition?
:STATus:QUESTionable:TELEcom:M34[:EVENT]?
:STATus:QUESTionable:TELEcom:M34:CONDition?
:STATus:QUESTionable:TELEcom:M34:ENABle
:STATus:QUESTionable:TELEcom:M34:ENABle?
:STATus:QUESTionable:TELEcom:M34:PTRansition
:STATus:QUESTionable:TELEcom:M34:PTRansition?
:STATus:QUESTionable:TELEcom:M34:NTRansition
:STATus:QUESTionable:TELEcom:M34:NTRansition?
:STATus:QUESTionable:TELEcom:M8[:EVENT]?
:STATus:QUESTionable:TELEcom:M8:CONDition?
:STATus:QUESTionable:TELEcom:M8:ENABle
:STATus:QUESTionable:TELEcom:M8:ENABle?
:STATus:QUESTionable:TELEcom:M8:PTRansition
:STATus:QUESTionable:TELEcom:M8:PTRansition?
:STATus:QUESTionable:TELEcom:M8:NTRansition
:STATus:QUESTionable:TELEcom:M8:NTRansition?
:STATus:QUESTionable:TELEcom:M2[:EVENT]?
:STATus:QUESTionable:TELEcom:M2:CONDition?
:STATus:QUESTionable:TELEcom:M2:ENABle
:STATus:QUESTionable:TELEcom:M2:ENABle?
:STATus:QUESTionable:TELEcom:M2:PTRansition
:STATus:QUESTionable:TELEcom:M2:PTRansition?
:STATus:QUESTionable:TELEcom:M2:NTRansition
:STATus:QUESTionable:TELEcom:M2:NTRansition?

```

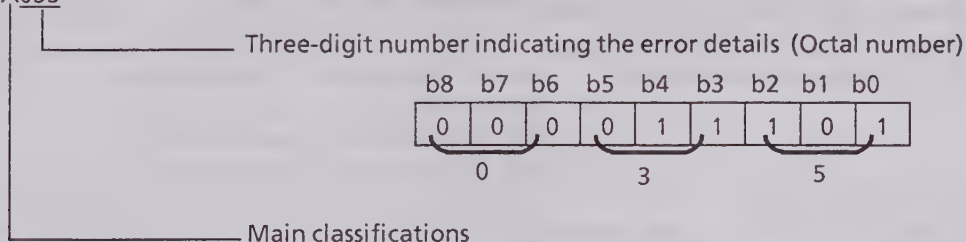
APPENDIX B

SELF-TEST ERROR CODES

The self-test for this instrument allows an error code to be displayed which corresponds to the type of error occurrence.

The error code comprises one alphabetic character and a three-digit number (Octal number).

Error code: A035



The table below shows the bit allocation for the main classifications of error codes.

Table B-1 Self-Test Error Codes (1/2)

Symbol	Main classifications	Detailed items corresponding to bits
A	139M signal	b0: Bit rate error ± 10 ppm or more
B	34M signal	b1: PRBS11 causes error/alarm.
C	8M signal	b2: PRBS15 causes error/alarm.
D	2M signal (unbalance)	b3: PRBS23 causes error/alarm.
E	2M signal (Balanced)	b4: Word causes error/alarm.
		b5: Frame on causes error/alarm.
F	Frequency offset option	b0: At 139M, error ± 5 ppm or more b1: At 34M, error ± 5 ppm or more b2: At 8M, error ± 5 ppm or more b3: At 2M, error ± 5 ppm or more
G	MUX option	b0: In MUX/DEMUX, error/alarm occurs.
H	Error insertion/detection	b0: Abnormal bit error insertion b1: Abnormal bit error detection b2: Abnormal frame error detection b3: Abnormal CRC-4 error detection b4: Abnormal CMI error insertion/detection b5: Abnormal bipolar error insertion/detection
I	Alarm insertion/detection 1	b0: Abnormal AIS 139M b1: Abnormal AIS 34M b2: Abnormal AIS 8M b3: Abnormal AIS 2M b4: Abnormal RA 139M b5: Abnormal RA 34M b6: Abnormal RA 8M b7: Abnormal RA 2M b8: Abnormal RA MF
J	Alarm insertion/detection 2	b0: Abnormal Signal Loss b1: Abnormal LOF 139M b2: Abnormal LOF 34M b3: Abnormal LOF 8M b4: Abnormal LOF 2M b5: Abnormal LOF MF b6: Abnormal Pattern sync Loss b7: Abnormal FAS alarm
K	Jitter modulation/measurement	b0: Jitter modulation or measurement is abnormal at 139 Mb/s. b1: Jitter modulation or measurement is abnormal at 34 Mb/s. b2: Jitter modulation or measurement is abnormal at 8 Mb/s. b3: Jitter modulation or measurement is abnormal at 2 Mb/s.
L	Jitter tolerance	b0: Tolerance is abnormal at 139 Mb/s. b1: Tolerance is abnormal at 34 Mb/s. b2: Tolerance is abnormal at 8 Mb/s. b3: Tolerance is abnormal at 2 Mb/s.

Table B-1 Self-Test Error Codes (2/2)

Symbol	Main classifications	Detailed items corresponding to bits
M	Jitter measurement frequency range	b0: Measurement frequency range is abnormal at 139 Mb/s. b1: Measurement frequency range is abnormal at 34 Mb/s. b2: Measurement frequency range is abnormal at 8 Mb/s. b3: Measurement frequency range is abnormal at 2 Mb/s.
N	Hit measurement	b0: Hit measurement is abnormal at 139 Mb/s. b1: Hit measurement is abnormal at 34 Mb/s. b2: Hit measurement is abnormal at 8 Mb/s. b3: Hit measurement is abnormal at 2 Mb/s.

(Blank)

APPENDIX C

SCPI ERROR MESSAGES

The response to the :SYSTem:ERRor? query SCPI command has a message with a code corresponding to an error defined by the SCPI standard.

The details of the error messages which this instrument supports are explained on the pages that follow.

C.1 Command Errors

An error code in the range [-199, -100] indicates that an IEEE488.2 syntax error has been detected. The occurrence of any error at this time causes bit 5 in the Event Status Register to be set. An error has been detected when one of the following events has occurred.

- A device has received a message which does not conform to the IEEE488.2 standard.
- A device has received an incorrect header which does not conform to the requirements for Device-dependent commands and common commands
- A Group Execute Trigger (GET) was received within a program message.

Table C-1 Error Messages for Command Errors (1/2)

Code	Message	Error-detecting conditions
-101	Invalid character	A invalid character is contained in a header or a parameter. Example: A header contains #. :SENSe:TELEcom:#RATE M139 A parameter contains \$. :SENSe:TELEcom:PATTeRn:TYPE PRBS11
-104	Data type error	The parameter type is different from the specified type. Example: Character data is sent to a command whose data is numeric.
-105	GET not allowed	A group Execute Trigger (GET) is received within a program message.
-108	Parameter not allowed	The number of parameters received is more than is specified. Example: Four parameters are sent where two parameters are specified. :SENSe:TELEcom:DEMuX:ROUte M34,1,M8,2
-112	Program mnemonic too long	A program mnemonic contains more than twelve characters. Example: More characters are appended because the header colon is missing. :SENSe:TELEcom:PERfORManceTYPE G821
-113	Undefined header	The header is syntactically correct, but it is undefined for this specific device. Example: The header spelling is incorrect. :SENSe:TELEcom:DEMuX:ROOT M34,1
-120	Numeric data error	There are errors in the numeric data.

Table C-1 Error Messages for Command Errors (2/2)

Code	Message	Error-detecting conditions
-121	Invalid character in number	Numeric data contains an invalid character. Example: Numeric data contains a symbol character. :SENSe:TELEcom:M2:MCHanne1 3\$
-130	Suffic error	There are suffix errors. Example: Part of the data was written in Roman characters by mistake instead of numeric parameter data. :SENSe:TELEcom:M2:MCHanne1 3i
-144	CHaracter data too long	Character data contains more than twelve characters.
-150	String data error	A string data is unstipulated. Example: Single quotation marks and double quotation marks are mixed. :DISPlay:MENU:NAME "SYSTem"

C.2 Execution Errors

An error code in the range [-299, -200] indicates that an error has been detected in the instrument's execution control block. The occurrence of any error at this time causes bit 4 in the Event Status Register to be set.

An error has been detected when one of the following events has occurred.

- <PROGRAM DATA> following a header cannot be accepted which exceeds the instrument's capabilities.
- A program message cannot be executed due to the condition of the instrument.

Table C-2 Error Messages for Execution Errors (1/2)

Code	Message	Error-detecting conditions
-220	Parameter error	There are errors in the parameter elements.
-221	Setting conflict	A parameter is correct but cannot be set due to the state of the instrument. Example: When an attempt is made to execute a self-test during measurement, a bit rate of higher speed than the interface is specified for the last stage of MUX.
-222	Data out of range	Numeric data is out of the range which can be accepted by the instrument. Example: :SENSe:MEASure:PERiod 0, S :SYSTem:MEMory:STORe 25
-223	Too much data	A longer data string is received than is stipulated. Example: Although the word length to be set is 4 bits, a 5-bit word has been set as shown below. :SENSe:TELEcom:PATtern:UWORD "11001"

Table C-2 Error Messages for Execution Errors (2/2)

Code	Message	Error-detecting conditions
-224	Illegal parameter value	The accepted parameter cannot be executed due to invalid parameter values. Example: Character data is unstipulated. :SENSe:TELEcom:BRAtE M156 Data string is unstipulated. :CALCulate:DATA? "LAST:EC:CMI"
-240	Hardware error	A valid command cannot be executed because of a hardware problem in the instrument.
-241	Hardware missing	A valid command cannot be executed because an option is not installed.

C.3 Device-Specific Errors

An error code in the range [-399, -300] indicates that the instrument has detected an error which is neither a command error, a query error, nor an execution error. A problem with the hardware, firmware, or self-test etc. causes a device-specific error. The occurrence of any error at this time causes bit 3 in the Event Status Register to be set.

Table C-3 Error Messages for Device-Specific Errors

Code	Message	Error-detecting conditions
-310	System error	A system error has occurred.
-311	Save/recall memory lost	The data saved in local memory has been lost.
-315	Configuration memory lost	The data saved in non-volatile memory has been lost.
-330	Self-test failed	The self-test has failed due to an abnormality.

C.4 Query Errors

An error code in the range [-499, -400] indicates that the output queue control of the instrument has detected an error due to the message exchange protocol. The occurrence of any error at this time causes bit 2 in the Event Status Register to be set.

An error has been detected when one of the following events has occurred.

- An attempt is made to read data in the Output Queue when there is none there.
- Data in the Output Queue has been lost.

Table C-4 Error Messages for Query Errors

Code	Message	Error-detecting conditions
-410	Query INTERRUPTED	The interrupted action is executed by a new command before the instrument finishes sending a response message. Example: *TRG is sent before a reply is sent fully.
-420	Query UNTERMINATED	The controller attempts to read a response message without having sent the query command corresponding to the message or after having sent an unterminated query command.
-430	Query DEADLOCKED	An attempt is made to buffer more data than there is space for. Example: The instrument cannot continue measurement because both input buffer and output buffer are full.

APPENDIX D

CORRESPONDENCE BETWEEN SCREENS AND COMMANDS

This appendix describes the correspondence between program commands and panel display screens. For details of program commands, see paragraph 9.3 Device-dependent Commands.

D.1 The Correspondence between Condtn Screens and Commands

D.1.1 Condtn:Tx&Rx screen

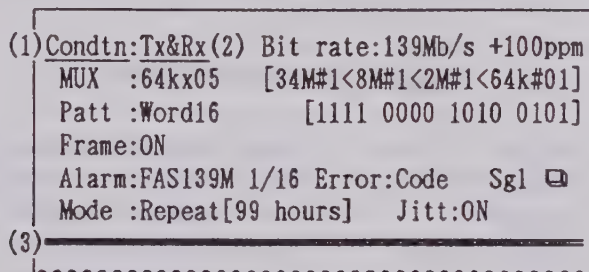


Fig. D-1 Condtn:Tx&Rx Screen

- (1) Switching to the specified screen
:DISPlay:MENU[:NAME]
- (2) Settings common to both transmitter and receiver sections
:INSTrument:COUPlE
- (3) Querying measurement state
:SENSe:MEASure:STATe?

D.1.2 Condtn:Tx screen

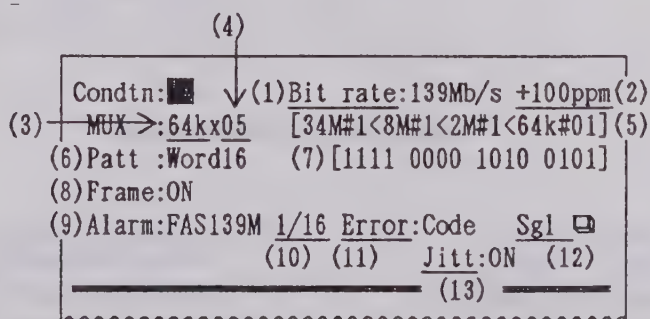


Fig. D-2 Condtn:Tx Screen

- | | |
|------------------------------------|-----------------------------------|
| (1) :SOURce:TELecom:BRATE | (7) :SOURce:TELecom:PATTErn:UWORD |
| (2) :SOURce:TELecom:OFFSet | (8) :SOURce:TELecom:FRAMing |
| (3) :SOURce:TELecom:MUX:MRATE | (9) :SOURce:TELecom:ALARM:TYPE |
| (4) :SOURce:TELecom:MUX:N | (10) :SOURce:TELecom:ALARM:ARATE |
| (5) :SOURce:TELecom:MUX:ROUTE | (11) :SOURce:TELecom:ERRor:TYPE |
| (6) :SOURce:TELecom:PATTErn[:TYPE] | (12) :SOURce:TELecom:ERRor:ERATE |
| | (13) :SOURce:TELecom:JITTer |

D.1.3 Condtn:Rx screen

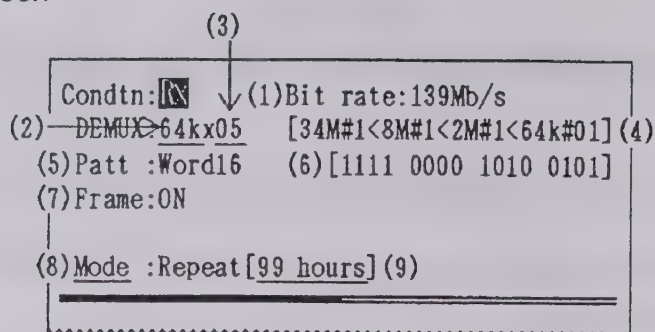


Fig. D-3 Condtn:Rx Screen

- | | |
|--------------------------------|-----------------------------------|
| (1) :SENSe:TELecom:BRATE | (5) :SENSe:TELecom:PATtern[:TYPE] |
| (2) :SENSe:TELecom:DEMuX:MRATE | (6) :SENSe:TELecom:PATtern:UWORD |
| (3) :SENSe:TELecom:DEMuX:N | (7) :SENSe:TELecom:FRAMing |
| (4) :SENSe:TELecom:DEMuX:ROUTE | (8) :SENSe:MEASure:TYPE |
| | (9) :SENSe:MEASure:PERiod |

D.2 The Correspondence between Results Screens and Commands

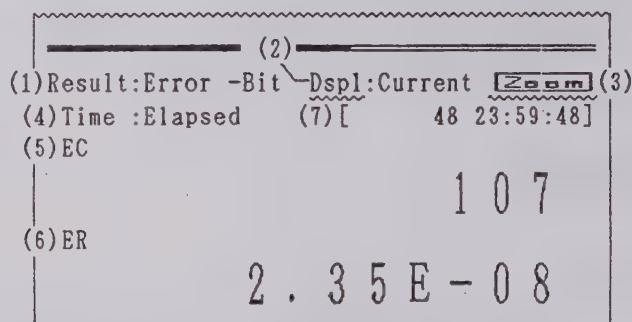


Fig. D-4 Results Screen

- | | |
|--------------------------------|---------------------------|
| (1) :DISPlay:RESult:TYPE | (7) Displaying the time |
| (2) :DISPlay:RESult:MODE | :SENSe:MEASure:ELAPsed? |
| (3) :DISPlay:RESult:ZOOM:SET | :SENSe:MEASure:TIMed? |
| (4) :DISPlay:RESult:TIME | (8) Displaying the result |
| (5) :DISPlay:RESult:ZOOM:ITEM1 | :CALCulate:DATA? |
| (6) :DISPlay:RESult:ZOOM:ITEM2 | |

D.3 The Correspondence between Analysis Screens and Commands

D.3.1 Analysis:Trace Text screen

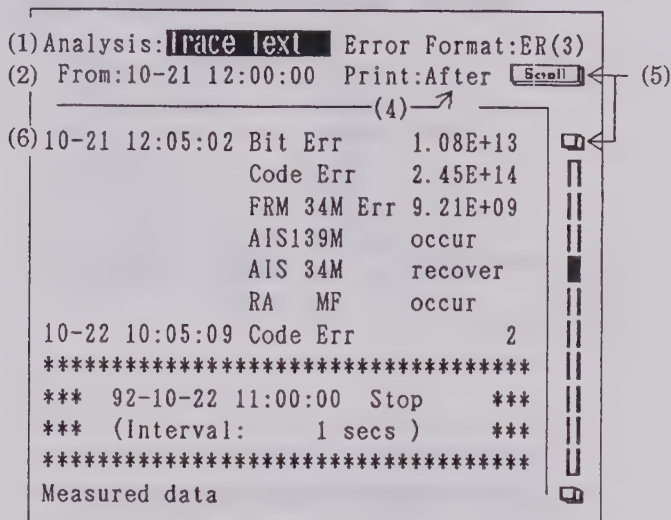


Fig. D-5 Analysis:Trace Text Screen

- | | |
|---|---|
| (1) Switching to the specified screen
:DISPlay:MENU[:NAME] | (6) Displaying the log data
:CALCulate:LOG:DATA? |
| (2) :DISPlay:ANALysis:TTEXT:FROM | Selecting the display item |
| (3) :DISPlay:ANALysis:TTEXT:FROMat | :CALCulate:LOG:FORMAT |
| (6) :DISPlay:ANALysis:TTEXT:PRINT | |
| (7) :DISPlay:ANALysis:TTEXT:SCROLL | |

D.3.2 Analysis:Trace Graph screen

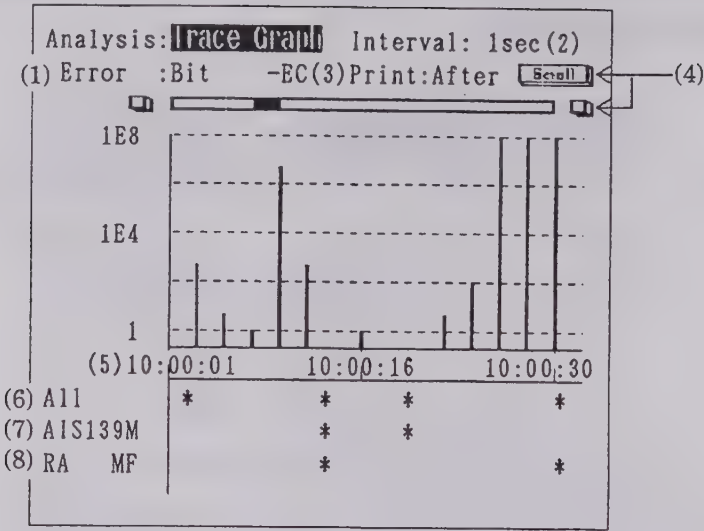


Fig. D-6 Analysis:Trace Graph Screen

- (1) :DISPLay:ANALySis:TGRaph:ERRor
- (2) :DISPLay:ANALySis:TGRaph:INTerval
- (3) :DISPLay:ANALySis:TGRaph:PRINT
- (4) :DISPLay:ANALySis:TGRaph:SCRoLL
- (5) :DISPLay:ANALySis:TGRaph:FROM
- (6) :DISPLay:ANALySis:TGRaph:ALARm1
- (7) :DISPLay:ANALySis:TGRaph:ALARm2
- (8) :DISPLay:ANALySis:TGRaph:ALARm3

D.3.3 Analysis:Distribution screen

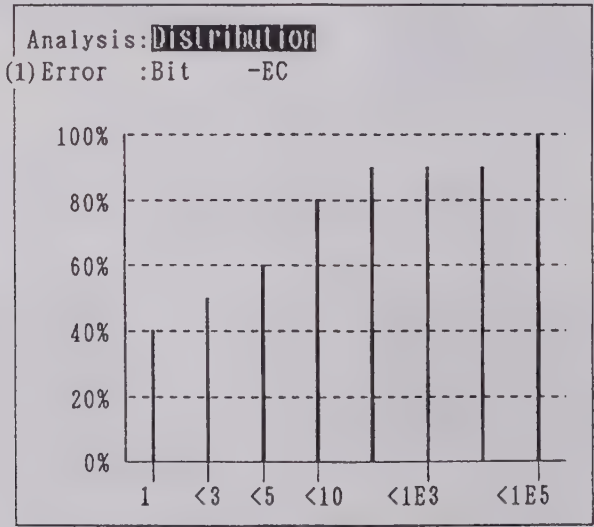


Fig. D-7 Analysis:Distribution Screen

- (1) :DISPLay:ANALySis:DISTRibution:ERRor

D.4 The Correspondence between Setup Screens and Commands

D.4.1 Setup:Measure screen

```

(1) Setup: Measure

(2) Measuring mode      : In-service
(3) Input mode         : Term
(4) 2M signal input    : Unbalanced
    (5) CRC-4          : ON
    (6) Channels        : 30
    (7) Signalling      : ON
(8) Program start      : ON [10-01 10:00:00] (9)
  
```

Fig. D-8 Setup:Measure Screen

- | | |
|---|---------------------------------|
| (1) Switching to the specified screen
:DISPlay:MENU[:NAME] | (5) :SENSe:TELEcom:M2:CRc |
| (2) :SENSe:TELEcom:MMODE | (6) :SENSe:TELEcom:M2:MCHannel |
| (3) :SENSe:TELEcom:IMODE | (7) :SENSe:TELEcom:M2:SIGNaling |
| (4) :ROUTE:INPut:SElect | (8) :SENSe:MEASure:BTIME:SET |
| | (9) :SENSe:MEASure:BTIME:START |

D.4.2 Setup:Performance screen

```

Setup: Performance

Type          : G.821
Frame(DEMUX)  : 139M
Threshold {
  %ES : ON [12.3456%]
  %SES: OFF
  (1) %DM : ON [ 3.1111%]
  %US : OFF
}
  
```

Fig. D-9 Setup:Performance Screen (G.821)

- | |
|--|
| (1) :CALCulate:TELEcom:PERformance:THReshold:G821:ES:SET |
| :CALCulate:TELEcom:PERformance:THReshold:G821:SES:SET |
| :CALCulate:TELEcom:PERformance:THReshold:G821:DM:SET |
| :CALCulate:TELEcom:PERformance:THReshold:G821:US:SET |
| (2) :CALCulate:TELEcom:PERformance:THReshold:G821:ES:S1 |
| :CALCulate:TELEcom:PERformance:THReshold:G821:SES:S1 |
| :CALCulate:TELEcom:PERformance:THReshold:G821:DM:S1 |
| :CALCulate:TELEcom:PERformance:THReshold:G821:US:S1 |

Setup: Performance	
Type	:G. 826
Frame(DEMUX)	:139M
Threshold	ESR :ON [1.23E+06]
(1)	SES:OFF (2)
	BBER:OFF
	%US :OFF

Fig. D-10 Setup:Performance Screen (G.826)

- (1) :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:ESR:SET
 :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:SES:SET
 :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:BBER:SET
 :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:US:SET
 (2) :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:ESR:S1
 :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:SES:S1
 :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:BBER:S1
 :CALCulate:TELEcom:PERFORMANCE:THReshold:G826:US:S1

Setup: Performance	
(1)Type	:M. 2100
(2)Frame(DEMUX)	:139M
Threshold	ES :ON [S1:12345 S2:12346]
(3)	SES:OFF (4) (5)

Fig. D-11 Setup:Performance Screen (M.2100)

- (1) :CALCulate:TELEcom:PERFORMANCE:TYPE
 (2) :CALCulate:TELEcom:PERFORMANCE:FRAME
 (3) :CALCulate:TELEcom:PERFORMANCE:THReshold:M2100:ES:SET
 :CALCulate:TELEcom:PERFORMANCE:THReshold:M2100:SES:SET
 (4) :CALCulate:TELEcom:PERFORMANCE:THReshold:M2100:ES:S1
 :CALCulate:TELEcom:PERFORMANCE:THReshold:M2100:SES:S1
 (5) :CALCulate:TELEcom:PERFORMANCE:THReshold:M2100:ES:S2
 :CALCulate:TELEcom:PERFORMANCE:THReshold:M2100:SES:S2

D.4.3 Setup:Print screen

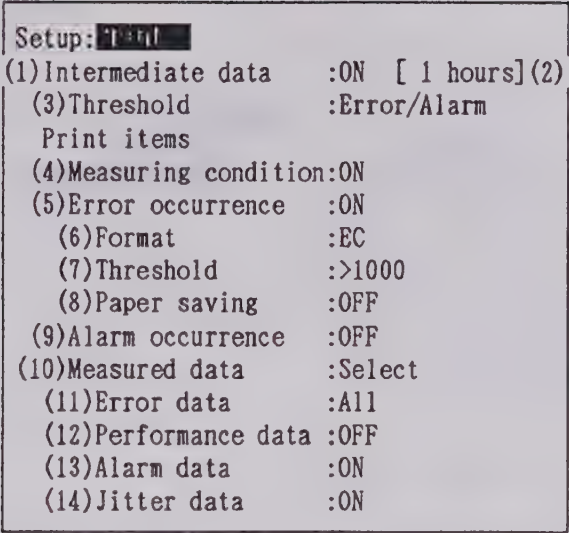


Fig. D-12 Setup:Print Screen

- | | |
|-----------------------------------|--------------------------------------|
| (1) :SYSTem:PRINt:IDATa[:TYPE] | (7) :SYSTem:PRINt:ERRor:THReshold:EC |
| (2) :SYSTem:PRINt:IDATa:PERiod | :SYSTem:PRINt:ERRor:THReshold:ER |
| (3) :SYSTem:PRINt:IDATa:THReshold | (8) :SYSTem:PRINt:ERRor:SQUelch |
| (4) :SYSTem:PRINt:MCONdition | (9) :SYSTem:PRINt:ALARm:SET |
| (5) :SYSTem:PRINt:ERRor:SET | (10) :SYSTem:PRINt:MDATa:TYPE |
| (6) :SYSTem:PRINt:ERRor:TYPE | (11) :SYSTem:PRINt:MDATa:ERRor |
| | (12) :SYSTem:PRINt:MDATa:PERFormance |
| | (13) :SYSTem:PRINt:MDATa:ALARm |
| | (14) :SYSTem:PRINt:MDATa:JITTer |

D.4.4 Setup:Memory screen

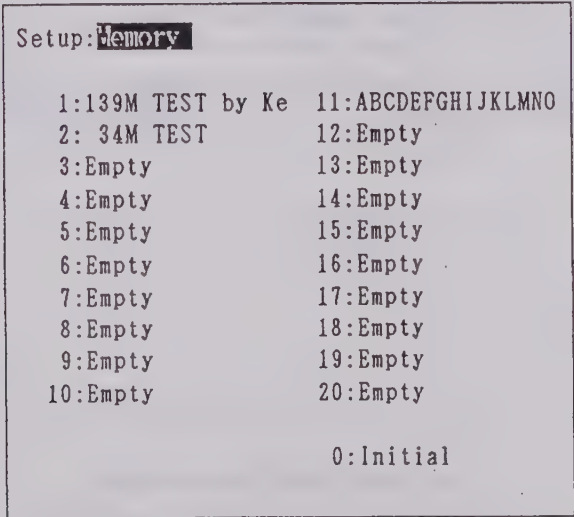


Fig. D-13 Setup:MemoryScreen

- | | |
|-----------------------|----------------------|
| :SYSTem:MEMory:RECall | :SYSTem:MEMory:CLEAr |
| :SYSTem:MEMory:STORE | :SYSTem:MEMory:LABEl |

D.5 The Correspondence between System Screen and Commands

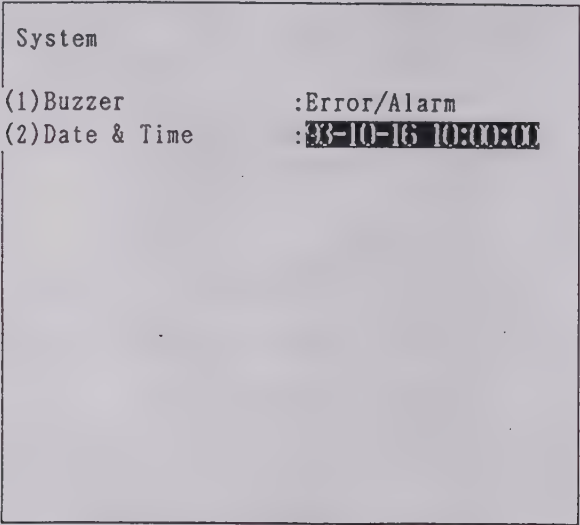


Fig. D-14 System Screen (When an external interface unit is not installed)

- (1) :SYSTem:BEEPer:TYPE
- (2) :SYSTem:DATE
:SYSTem:TIME

D.6 The Correspondence between Self-Test Screen and Commands

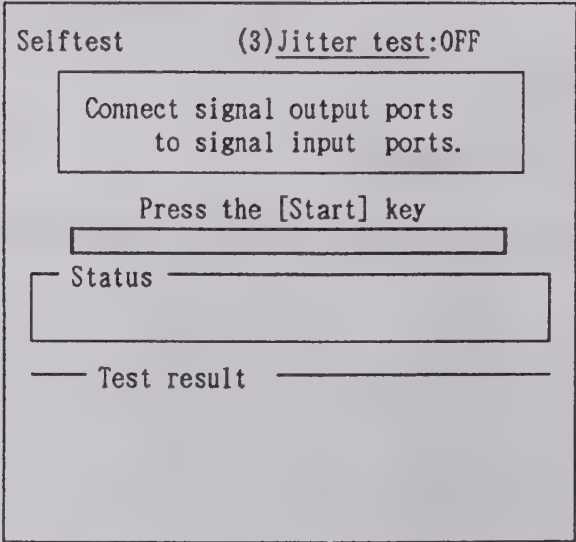


Fig. D-15 Self-Test Screen

- (1) Query for the test result
:TEST:DATA?
- (2) Query for self-test state
:TEST:STATE?
- (3) :TEST:JITTer

D.7 The Correspondence between Jitter Screen and Commands

D.7.1 Jitter:Manual screen

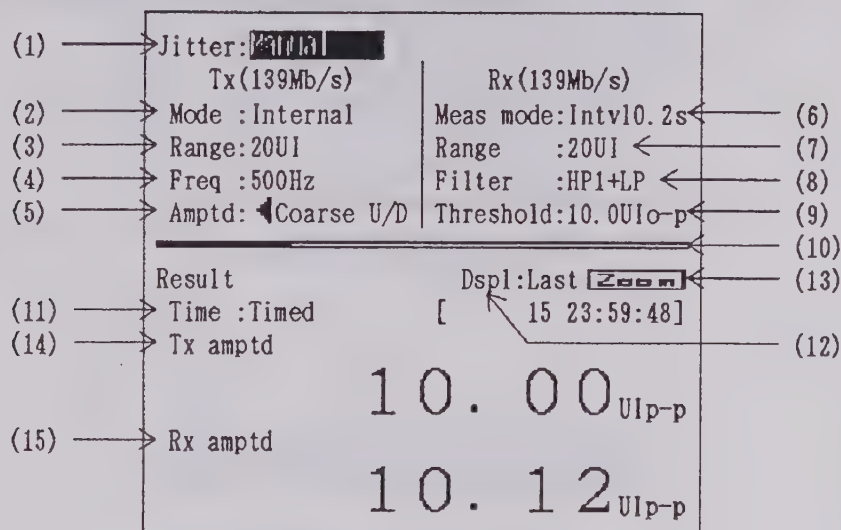


Fig. D-16 Jitter:Manual Screen

- | | |
|---|--|
| (1) Switching to the specified screen
:DISPlay:MENU[:NAME] | (8) :SENSe:JITTer:MANual:FILTer |
| (2) :SOURce:JITTer:MANual:MODE | (9) :SENSe:JITTer:MANual:THReshold |
| (3) :SOURce:JITTer:MANual:RANGe | (10) Querying measurement state
:SENSe:MEASure:STATe? |
| (4) :SOURce:JITTer:MANual:FREQuency | (11) :DISPlay:JITTer:MANual:TIME |
| (5) :SOURce:JITTer:MANual
:AMPLitude:STEP | (12) :DISPlay:JITTer:MANual:MODE |
| :SOURce:JITTer:MANual | (13) :DISPlay:JITTer:MANual:ZOOM:SET |
| :AMPLitude:DATA | (14) :DISPlay:JITTer:MANual:ZOOM:ITEM1 |
| (6) :SENSe:JITTer:MANual:MMode | (15) :DISPlay:JITTer:MANual:ZOOM:ITEM2 |
| (7) :SENSe:JITTer:MANual:RANGe | (16) Query measured result
:CALCulate:DATA? |

D.7.2 Jitter:Sweep mask screen

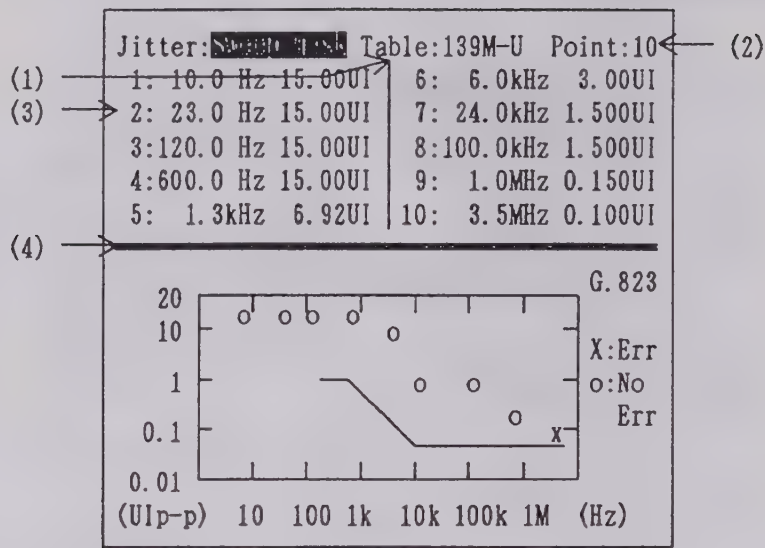


Fig. D-17 Jitter:Sweep mask Screen

- (1) :SOURce:JITter:SMASk:TABLE:TYPE

(2) :SOURce:JITter:SMASk:TABLE:POINT

(3) :SOURce:JITter:SMASk:TABLE:DATA
- (4) Querying Sweep mask measurement status
:SOURce:JITter:SMASk:STATe?

(5) Query Sweep mask measured result
:CALCulate:DATA?

D.7.3 Jitter:Transfer screen

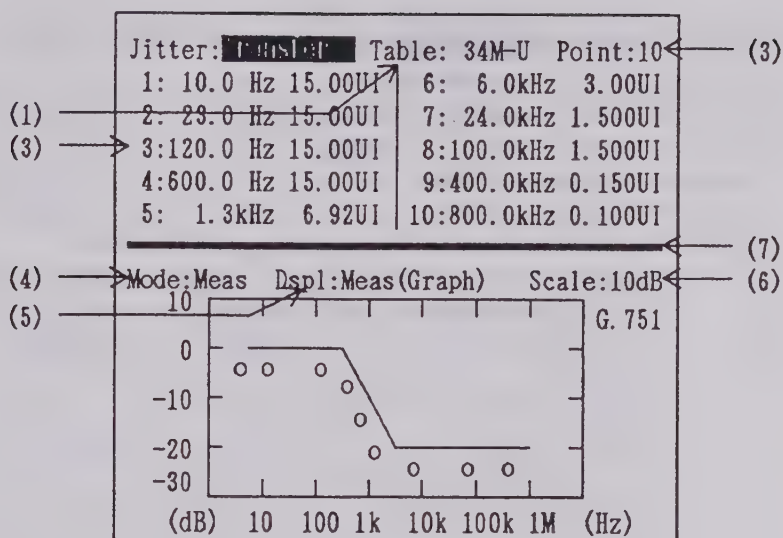


Fig. D-18 Jitter:Transfer Screen

- | | |
|---|--|
| (1) :SOURce:JITter:TRANsfer:TABLE:TYPE | (6) :DISPlay:JITter:TRANsfer:SCALE |
| (2) :SOURce:JITter:TRANsfer:TABLE:POINT | (7) Querying Transfer measurement status |
| (3) :SOURce:JITter:TRANsfer:TABLE:DATA | :SENSe:JITter:TRANsfer:STATE? |
| (4) :SENSe:JITter:TRANsfer:MODE | (8) Query Transfer measured result |
| (5) :DISPlay:JITter:TRANsfer:MODE | :CALCulate:DATA? |

D.7.4 Jitter:Jitt/Freq screen

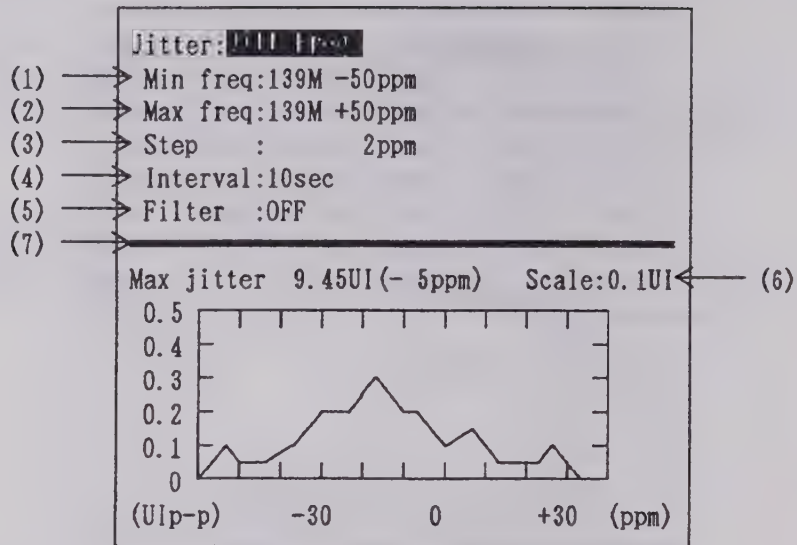


Fig. D-19 Jitter:Jitt/Freq Screen

- | | |
|--|---|
| (1) :SOURce:JITTer:JFRequency:OFFSet
:MIN | (6) :DISPlay:JITTer:JFRequency
:SCALe |
| (2) :SOURce:JITTer:JFRequency:OFFSet
:MAX | (7) Querying Jitt/Freq measurement state
:SENSe:JITTer:JFRequency
:STATe? |
| (3) :SOURce:JITTer:JFRequency:TABLE
:DATA | (8) Query Jitt/Freq measured result
:CALCulate:DATA? |
| (4) :SENSe:JITTer:JFRequency:MODE | |
| (5) :DISPlay:JITTer:JFRequency:MODE | |

D.8 The Correspondence Between Front Panel and Other Commands

Table D-4 The Correspondence Between Front Panel and Other Commands

Start/ Stop key	Measurement	:SENSe:MEASure:START :SENSe:MEASure:STOP
	Self-test	:TEST:START :TEST:STOP
	Sweep mask measurement	:SENSe:JITTer:SMASk:START :SENSe:JITTer:SMASk:STOP
	Transfer measurement	:SENSe:JITTer:TRANSfer:START :SENSe:JITTer:TRANSfer:STOP
	Jitt/Freq measurement	:SENSe:JITTer:JFRequency:START :SENSe:JITTer:JFRequency:STOP
Print key		:SYSTem:PRINt:ENABle
Copy key		:SYSTem:PRINt:COPY
Paper Feed key		:SYSTem:PRINt:FEED
History Reset key		:SYSTem:LED:RESet
Others		:SYSTem:PRINt:TEXT :SYSTem:ERRor? :SYSTem:VERSion? STARus Subsystem

(Blank)

APPENDIX E

DETAILED FUNCTIONS OF INITIALIZATION COMMANDS

Appendix E describes the range in which the types of initialization commands affect a device.

E.1 Bus Initialization

IFC: All interface functions connected to the GPIB bus line are initialized by the IFC message. (When the GPIB option is installed)

Initializing the interface functions allows release of the controller-set device's interface functions to reset them to their initial state.

In the table below, ○ indicates the functions which are initialized; △ indicates the functions which are partially initialized.

This command does not affect the operation status of a device.

Table E-1 State transition of the interface function by the IFC

No.	Function	Code	Initialization by IFC
1	Source handshake	SH	○
2	Acceptor handshake	AH	○
3	Talker or extended talker	T or TE	○
4	Listener or extended talker	L or LT	○
5	Service request	SR	△
6	Remote/local	RL	
7	Parallel poll	PP	
8	Device clear	DC	
9	Device trigger	DT	
10	Controller	C	○

The following shows the state transition of the device affected by the IFC message.

Table E-2 State Transition of a Device by the IFC Message

Talker/listener	All talkers and listeners are put in the idle state (TIDS, LIDS) within 100 μ s.
Controller	The controller is put in the idle state (CIDS-Controller Idle State) within 100 μ s if it is not active (SACS-System Control Active State).
Service request device	The IFC message does not affect the state (the SRQ line has been set to LOW level by a device) in which a device is sending an SRQ message to the controller, but releases the state into which the controller has put all devices under the system bus, into serial poll mode.
Remote devices	The IFC message does not release a device which has been set to remote mode.

E.2 Message Exchange Initialization

DCL, SDC: Executes initialization for message exchange of all devices on the GPIB bus or the device whose address is specified. (When the GPIB option is installed)

The table below shows the items to be initialized for message exchange.

Table E-3 The Items to be Initialized for Message Exchange.

Input Buffer and Output Queue	Cleared
Parser, Execution Controller, and Response Formatter	Reset
Device commands including *RST	Clears all commands that prevent the execution of these commands.
Coupled-parameter program messages	Discards all commands and queries deferred due to coupled parameters.
Processing the *OPC command	Puts a device in OCIS (operation Complete Idle State) state. As a result, the operation complete bit cannot be set in the Standard Event Status Register.
Processing the *OPC? query	Puts a device in OQIS (Operation Complete Query Idle State) state. As a result, the operation complete bit cannot be set in the Output Queue. The MAV bit is cleared.
Device functions	Functions for message exchange are put in the idle state. The device continues to wait for a message from the controller.

E.3 Device Initialization

***RST:** Initializes the device (common to both GPIB and RS-232C option)

The ***RST** command causes a device to be switched to a different state as shown below.

1. Returns the Device-dependent functions to a known state that is independent of the past-use history of the device.
This instrument is returned to the state it was in when the power was turned off, immediately after the power is turned back on.
2. Puts a device in OCIS (Operation Complete Idle State) state.
3. Puts a device in OQIS (Operation Complete Query Idle State) state.

The ***RST** command does not affect the items listed below.

1. IEEE488.2 interface stat
 2. Device address
 3. Output Queue
 4. Service Request Enable Register
 5. Standard Event Status Enable Register
 6. Power-on-status-clear flag setting
 7. Calibration data affecting device specification
 8. Macros defined by DMC command
 9. Response messages for the PUD query
 10. Response messages for the RDT query
- (Items 8 through 10 are not supported by this instrument)

E.4 Device Status at Power-on

When the device is powered on:

1. The device state is as it was when the power was last switched off.
2. The Input Buffer and Output Queue are cleared.
3. The Parser, Execution Control, and Response Formatter are reset.
4. The device is put in OCIS state.
5. The device is put in OQIS state.
6. If the PSC flag is TRUE, a Event Status Enable Registers are cleared. Events can be recorded after the registers have been cleared.

Items related to power-on-status-clear (PSC) flag

When the PSC flag is FALSE, the Service Request Enable Register, Standard Event Status Enable Register, and other Status Enable Registers are not affected. When the PSC flag is TRUE or the ***PSC** command is not being executed, these registers are cleared.

(Blank)

Anritsu Service and Sales offices

America

Anritsu Wiltron Company

685 Jarvis Drive, Morgan Hill, CA
95037-2809, U.S.A
Phone: 408-776-8300
Fax: 408-776-1744

New Jersey Office

10 Kingsbridge Road Fairfield, NJ
07004, U.S.A
Phone: 201-227-8999
Fax: 201-575-0092

Anritsu Wiltron Eletronica Ltda.

Praia de Botafago 440, Sala 2401
CEP 22250-040, Rio de Janeiro, RJ,
Brazil
Phone: 021-286-9141
Fax: 021-537-1456

Anritsu Wiltron Instruments Ltd.

Unit 102, 215 Stafford Road Nepean,
Ontario K2H 9C1, Canada
Phone: 613-828-4090
Fax: 613-828-5400

Electronica 2000, S.A. de C.V.

Mexico, Mexico
Phone 05-662-8800/9989

Electronic Engineering S.A.

San Jos , Costa Rica
Phone: 2258793/2533759

HDM Elquitecnica Cia. Ltda.

Quito, Ecuador
Phone: 02-464587

Iota S.A.

Buenos Aires, Argentina
Phone: 01-432-8522/8459

Parker S.A.

Santiago, Chile
Phone: 02-231-9976/7147

Radiocom S.A.

Bogota, Colombia
Phone: 01-218-2054

Radiocomunicaciones cruz C.A.

Caracas, Venezuela
Phone: 02-7932322

Suministros Industriales, S.A.

Panama City, Panama
Phone: 64-2605, 69-2738

Vernal S.A.

Montevideo, Uruguay
Phone: 02-920171/920301

Europe

Anritsu Wiltron AB

Box 247, Jägerhorns väg 19, S-127
25 Skärholmen, Sweden
Phone: 08-7405840
Fax: 08-7404800

Anritsu Wiltron GmbH

Grafenberger Allee 54-56, D-40237 D
sseldorf 1, Germany
Phone: 0211-679760
Fax: 0211-683353

Anritsu Wiltron Ltd.

Capability Green, Luton, Bedfordshire
LU1 3LU, United Kingdom
Phone: 1582-418853
Fax: 1582-31303

Anritsu Wiltron S.A.

9, avenue du Québec ZAC de
Courtaboeuf 91951 LES ULIS
CEDEX, France
Phone: 01-64 46 65 46
Fax: 01-64 46 10 65

Anritsu Wiltron S.p.A.

Via Elio Vittorini, N.129, 00144 Roma
Eur, Italy
Phone: 06-5022666
Fax: 06-5022425

Anritsu Wiltron S.p.A. Milano Office

20041 Agrate Brianza (MI), C.D.
Colleoni, Via Paracelso 14, Italy
Phone: 039-639411
Fax: 039-639419

C.N. Rood B.V.

Rijswijk, Netherlands
Phone: 0-70-399-6360

C.N. Rood S.A./N.V.

Zellik, Belgium
Phone: 02-466-8199

Elsinco Bratislava s. r. o.

Bratislava, Slovakia
Phone: 0222-8121751

Elsinco Budapest KFT

Budapest, Hungary
Phone: 01-112-4854

Elsinco GmbH

Wien, Austria
Phone: 0222-8121751

Elsinco Polska Sp.z.o.o.

Warazawa, Poland
Phone: 022-396979

Elsinco Praha s. r. o.

Praha, Czecho
Phone: 02-4702 ex 451, 453

Elsinco Sofia

Sofia, Bulgaria
Phone: 02-581-698

G'Amundason H.F.

Reykjavik, Iceland
Phone: 01-687820

GMP S.A.

Lausanne, Switzerland
Phone: 021-6348181

Instrutek Periferi A/S

Horsens, Denmark
Phone: 75-611100

Anritsu Wiltron

Kappelitie 8, 02200 Espoo, Finland
Phone: 0-5259-1890
Fax: 0-4522-344

Kostas Karayannis S.A.

Athens, Greece
Phone: 01-323-0303/7731

Omnitecnica

Amadora, Portugal
Phone: 01-4715517

Teleinstrument AS

Asker, Norway
Phone: 66-901190

Unitronics, S.A.

Madrid, Spain
Phone: 91-542 52 04

Wien Schall GesmbH

Wien, Austria
Phone: 1-81155-140

Asia, Pacific and Africa

Alcatel Australia Ltd.

Southern Melbourne, Australia
Phone: 03-694-6666
Alcatel Papua New Guinea Ltd.
Port Moresby, Papua New Guinea
Phone: 675-256933

**Anritsu Wiltron (Singapore)
Pte Ltd.**

3 Shenton Way, #24-03, Singapore
106
Phone: 2265206
Fax: 2265207

**Anritsu Wiltron Company
Limited**

Suite 812, 8/F., Chinachem Golden
Plaza, 77 Mody Road, Tsim Sha Tsui
East Kowloon,
Hong Kong
Phone: 2301-4980
Fax: 2301-3545

**Associated Electric Trading
Corp.**

Lahore, Pakistan
Phone: 042-722-1716/7857

Beijing Liaison Office

Room No. 416, Beijing Fortune
Building 5 Dong San Huan Bei-lu,
Chaoyang-qu Beijing 100004, China
Phone: 861-501-7551, 7552
Fax: 861-501-7553

**Cha wei Electric Trading
Co., Ltd.**

Taipei, Taiwan
Phone: 02-561-2211

Chris Radiovision Ltd.

Nicosia, Cyprus
Phone: 02-466121

Etecsa CC

Midrand, South Africa
Phone: 11-315-1366

**Giza Systems Engineering
Co.**

Cairo, Egypt
Phone: 02-3490140

**Han-Il Commerce & Com-
pany**

Seoul, Korea
Phone: 02-783-6391

Infotechs Ltd.

Colombo, Sri Lanka
Phone: 01-580088/500817

**Inter Mühendislik
Danismanlik ve Ticaret A.S.**

Istanbul, Turkey
Phone: 216-349 9400

Meera Agencies (P) Ltd.

New Delhi, India
Phone: 685-3228/3959

**Mercantile Information
Telecommunication Tech-
nology Est.**

Abu Dhabi, U.A.E.
Phone: 2-768909

**National Projects and
Technology Co. Ltd**

Wadi Kabir, Sultanate of Oman
Phone: 791704

New Delhi Liasison Office

Room No. 508, Prakash Deep 7,
Tolstoy Marg, New Delhi-110001,
India
Phone: 011-3319133
Fax: 011-3713948

**O'Connor's Engineering
and Trading (Malaysia) Bhd**

Selangor, Malaysia
Phone: 03-757-2828

**O'Connor's Singapore Pte
Ltd.**

Singapore
Phone: 473-7944

Schmidt Telecom

Hong Kong
Phone: 507-0222

Scollay Holding Ltd.

Wellington, New Zealand
Phone: 04-4-722-961

P & P Telecom Co., Ltd.

Bangkok, Thailand
Phone: 02-271-2397/0057

P.T. Subur Sakti Putera

Jakarta, Indonesia
Phone: 021-3858361

Rajab & Silsilah & Co.

Jeddah, Saudi Arabia
Phone: 2-6610006

**Salritsu International Trad-
ing Corporation**

Metro Manila, Philippines
Phone: 632-8150946/8938998

Uto Trading Company

Seoul, Korea
Phone: 02-765-1800/1801

Anritsu Wiltron Corporation

Room No. 901 Daeo Bldg., 26-5,
Yeoido-Dong, Young Deung Po-Ku,
Seoul, Korea
Phone: 2-782-7151~6
Fax: 2-782-4590

Japan

Head Office

5-10-27, Minamiazabu, Minato-ku,
Tokyo 106
Phone: 03-3446-1111
Fax: 03-3442-0235

Atsugi Factory

1800, Onna, Atsugi-si, Kanagawa 243
Phone: 0462-23-1111
Fax: 0462-25-8379

Koriyama Factory

301, Dooba, Aza, Koriyama-shi,
Fukushima 963
Phone: 0249-56-2112
Fax: 0249-56-2299

Osaka (Anritsu Technics)

1-10-31, Mikunihonmachi,
Yodogawa-ku, Osaka 532
Phone: 06-391-0697
Fax: 06-391-1748



ANRITSU CORPORATION

5-10-27, Minamiazabu, Minato-ku, Tokyo, 106 Japan

PHONE. 03-3446-1111

TELEX J 34372 ANRITU